

①

Computing Science and Statistics

Volume 24

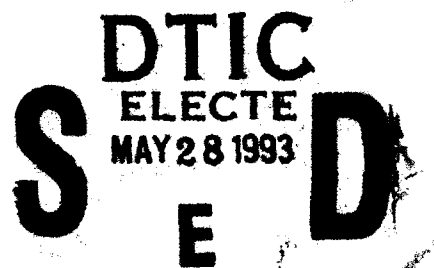


AD-A265 181



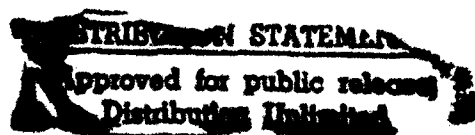
Graphics and Visualization

H. Joseph Newton
Editor



Proceedings of the
24th Symposium on the Interface
College Station, Texas, March 18-21, 1992

INTERFACE
FOUNDATION
OF NORTH AMERICA



Computing Science and Statistics

Volume 24

Graphics and Visualization

Editor

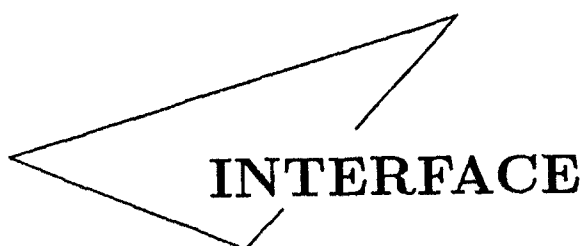
H. Joseph Newton

Texas A & M University, College Station, Texas

Proceedings of the
24th Symposium on the Interface

DTIC QUALITY INSPECTED 6

Accession For	
NTIS CRA&I	<input checked="" type="checkbox"/>
DTIC TAB	<input type="checkbox"/>
Unannounced	<input type="checkbox"/>
Justification	
By	
Distribution /	
Availability Codes	
Dist	Avail and/or Special
A-1	



INTERFACE FOUNDATION OF NORTH AMERICA

93 5 27 03 5

6588 93-12032

PUBLISHER'S FOREWORD

This volume is the second published by the Interface Foundation of North America. In the past the volumes had been published by commercial publishers, but we had experienced substantial difficulties with availability, timeliness and cost. The publishing experience is an exciting new venture for the Interface Foundation. As many reader's of this volume will already know, the Interface Foundation has launched on a joint venture with the American Statistical Association and the Institute of Mathematical Statistics in the publication of the *Journal of Computational and Graphical Statistics*. We strongly encourage readership of and subscriptions to the new journal. The Interface series of meetings has been positioned squarely at the junction of computing science and statistics. Since the first Interface Proceedings was published in 1971, it has been the periodical of record in documenting the development of ideas at this junction. In 1987, the proceedings volume was re-titled with the main title being *Computing Science and Statistics* and with *Proceedings* being reserved as a sub-title. *Computing Science and Statistics* has had a much more archival nature than found in the typical conference proceedings. In keeping with this more archival nature, we have dropped the *Proceedings* subtitle and added a volume number to reflect the periodical character of the publication. Concomitant with this change, we suggest a change in the nature of the citation. An example of the recommended citation is as follows:

Eubank, R. L. and Speckman, P. L. (1992), "Practical simultaneous nonparametric regression confidence bands," *Computing Science and Statistics*, 24, 1-9.

Should more details be desired, the editor, the publisher (Interface Foundation of North America, Inc.) and the city (Fairfax Station, VA) may be added. The 1992 Interface Symposium was the 24th meeting. There were no proceedings of the first three meetings. We have chosen to number the volumes according to the meetings, rather than using a strict sequential numbering. This is intended to make the volume numbering somewhat easier to remember.

The papers and discussions in this volume are reproduced as received from authors. These presentations are presumed to be essentially as given at the 24th Symposium on the Interface. Volume 24 is not copyrighted by the Interface Foundation of North America. Publication in this volume does not preclude publication elsewhere.

AVAILABILITY OF PREVIOUS VOLUMES

23 (1991)	Interface Foundation of North America, Inc. P. O. Box 7460 Fairfax Station, VA 22039-7460
22 (1990)	Springer-Verlag New York, Inc. 175 Fifth Avenue New York, NY 10010
18, 19, 20, 21 (1986-1989)	American Statistical Association 1429 Duke Street Alexandria, VA 22314-3402

Volumes 20, 21, and 22 are also available from the Interface Foundation of North America, Inc. Interface, Interface '92, *Computing Science and Statistics*, and the triangle logo are trademarks of the Interface Foundation of North America, Inc.

PRINTED IN THE UNITED STATES OF AMERICA

REPORT DOCUMENTATION PAGE			Form Approved OMB No. 0704-0188	
Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington Headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188), Washington, DC 20503.				
1. AGENCY USE ONLY (Leave blank)	2. REPORT DATE 3/20/93	3. REPORT TYPE AND DATES COVERED Final Report 1 March 92-31 May 92		
4. TITLE AND SUBTITLE Computing Science and Statistics: Graphics and Visualization		5. FUNDING NUMBERS N00014-92-J-1459 R&T Code 4114623---01		
6. AUTHOR(S) N. Joseph Newton				
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Interface Foundation of North America, Inc. PO Box 7460 Fairfax Station, VA 22039-7460		8. PERFORMING ORGANIZATION REPORT NUMBER Vol. 24		
9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES) Office of Naval Research Code 1111 800 N. Quincy Street Arlington, VA 22217-5000		10. SPONSORING / MONITORING AGENCY REPORT NUMBER Office of Naval Research Resident Representative Suite 201 2135 Wisconsin Avenue NW Washington, D.C. 20007		
11. SUPPLEMENTARY NOTES				
12a. DISTRIBUTION / AVAILABILITY STATEMENT Distribution Unlimited			12b. DISTRIBUTION CODE	
13. ABSTRACT (Maximum 200 words) The 24th Symposium on the Interface: Computing Science and Statistics was held March 18-21, 1992 in College Station, Texas. The theme was Graphics and Visualization. Sixty invited papers were presented to more than 300 attendees. James Blinn gave the Keynote Address while Edward Tufte gave the Plenary Address. There were 38 technical sessions, including a poster session. There was also a short course on Visualization. This report presents papers based on the presentations. Also included is a list of attendees.				
14. SUBJECT TERMS Computational Statistics Scientific Visualization		Exploratory Data Analysis Multivariate Density Estimation Nonparametric Regression		15. NUMBER OF PAGES 626
				16. PRICE CODE
17. SECURITY CLASSIFICATION OF REPORT Unclassified	18. SECURITY CLASSIFICATION OF THIS PAGE Unclassified	19. SECURITY CLASSIFICATION OF ABSTRACT Unclassified		20. LIMITATION OF ABSTRACT

PREFACE

1992 Interface Proceedings

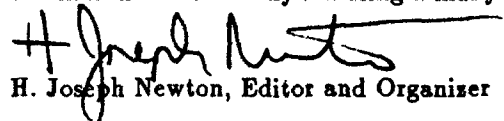
The 24th Symposium on the Interface of Computing Science and Statistics was held on March 18-21, 1992 at the College Station Hilton Hotel, College Station, Texas. The conference theme was "Graphics and Visualization."

The Department of Statistics at Texas A&M University hosted the meeting with H. Joseph Newton serving as Program Chair. The members of the program committee were Richard Becker, Mary Ellen Bock, Barbara Bottenfield, Alan Gelfand, James Gentle, James Hardin, Iain Johnstone, Robert Kass, Raoul LePage, Michael Longnecker, Douglas Martin, Gerald North, Emanuel Parzen, David Scott, Phil Spector, Michael Tarter, David Thomson, Dag Tjøstheim, Edward Wegman, Forrest Young, and Stuart Zweben.

The symposium theme was much in evidence in a variety of sessions, including the keynote address, "Making Computer Graphics Animations for Math and Science Education," by James Blinn of Caltech, a plenary address "Dequantification in Scientific Visualisation: Is this Science or Television?" by Edward Tufte of Yale, as well as a short course presented by Tufte based on his highly awarded books, "The Visual Display of Quantitative Information" and "Envisioning Information." Because of the highly visual nature of these talks, they could not be adequately presented in this proceedings.

Much of the success of a conference can be measured in terms of the number of attendees and the number of contributed talks, which, for this Symposium were approximately 305 and 77, respectively. Another indication of the lasting enthusiasm of the attendees is that fully 82% of the authors of contributed talks at the meeting have submitted manuscripts for this proceedings, while 67% of the invited speakers have submitted manuscripts. Many of the other invited speakers would have liked to submit papers, but their talks primarily consisted of showing videotapes of animated graphics and thus could not be presented in a traditional proceedings.

The organizer of an Interface Symposium truly has a number of people to thank. A special thanks goes to James Hardin for serving on the program committee as well as arranging and overseeing the vast array of computing and audio/visual equipment that was used during the meeting. Lorie Blackwell helped a great deal at the beginning of the planning process. Finally, I would like to especially thank Barbara Napoli Thomason for cheerfully handling a many details before, during, and after the symposium.



H. Joseph Newton, Editor and Organizer

HOST

Texas A&M University

COSPONSORS OF THE 24th SYMPOSIUM ON THE INTERFACE

National Security Agency

Office of Naval Research

Army Research Office

National Science Foundation

Air Force Office of Scientific Research

COOPERATING SOCIETIES AND INSTITUTIONS

American Statistical Association (ASA)

Institute of Mathematical Statistics (IMS)

Society for Industrial and Applied Mathematics (SIAM)

Operations Research Society of America (ORSA)

Texas A&M University

EXHIBITORS

IMSL, Inc.

Springer-Verlag, Inc.

Statistical Sciences, Inc.

SYSTAT, Inc..

Wadsworth & Brooks/Cole

John Wiley & Sons

SYMPOSIUM SCHEDULE

Wednesday, March 18, 1992

5:00 p.m.-8:00 p.m. Registration
5:00 p.m.-8:00 p.m. Board of Directors' Business Meeting and Dinner (Pecan)
8:00 p.m.-10:00 p.m. Opening Reception (Ballroom, I-III)

Thursday, March 19, 1992

8:15 a.m.-9:45 a.m. Keynote Address: "Designing Animations for Mathematics Education" (Ballroom, III-IV)
9:45 a.m.-10:00 a.m. Break (Ballroom V-VII)
10:00 a.m.-12:00 a.m. Invited A: Visualization Methods for Science and Statistics (Ballroom, III)
Invited B: Nonparametric Regression (Ballroom, IV)
Contributed A: Algorithms for Multivariate Distributions and Data (Ballroom, I)
Contributed B: Software Design and Quality (Ballroom, II)
12:00 p.m.-1:30 p.m. Lunch
1:30 p.m.-3:30 p.m. Invited A: Visualizing Programs (Ballroom, III)
Invited B: Spatial Time Series (Ballroom, IV)
Contributed A: Building on Existing Software (Ballroom, I)
Contributed B: Industrial Statistics (Ballroom, II)
3:30 p.m.-4:00 p.m. Break (Ballroom V-VII)
4:00 p.m.-6:00 p.m. Invited A: Visualizing Multivariate Data and Functions (Ballroom, III)
Invited B: Advanced Statistical Techniques for Industry (Ballroom, IV)
Contributed A: Approximating Integrals and Distributions (Ballroom, I)
Contributed B: Stochastic Processes and their Applications (Ballroom, II)
7:30 p.m.-10:30 p.m. Visualization Short Course—Edward Tufte (Brazos Amphitheater)

Friday, March 20, 1992

8:15 p.m.-9:45 p.m. Plenary Address: "Dequantification in Scientific Visualization: Is this Science or Television?" (Ballroom, III-IV)
9:45 a.m.-10:00 a.m. Break (Ballroom V-VII)
10:00 a.m.-12:00 p.m. Invited A: Statistical Visualization Software (Ballroom, III)
Invited B: Wavelets and Nonparametric Modeling (Ballroom, IV)
Contributed A: Linear Statistical Inference (Ballroom, I)
Contributed B: Applications of Graphics (Ballroom, II)
12:00 p.m.-1:30 p.m. Poster/Video/Demo Session (Ballroom V-VII)
1:30 p.m.-3:30 p.m. Invited A: Bayesian Computing (Ballroom, III)
Invited B: Statistical Methods in Software Quality Evaluation (Ballroom, IV)
Invited C: Building on S (Brazos Amphitheater)

Contributed A: Nonparametric Regression and Density Estimation, I (Ballroom, I)
Contributed B: Designing and Teaching Graphics (Ballroom, II)

3:30 p.m.-4:00 p.m. Break (Ballroom V-VII)

4:00 p.m.-6:00 p.m. Invited A: Geographic Information Systems (Ballroom, III)
Invited B: Sampling Based Approaches for Bayesian Inference (Ballroom, IV)
Invited C: Unix Tools for Statistical Computing (Brazos Amphitheater)
Contributed A: Nonparametric Regression and Density Estimation, II (Ballroom, I)
Contributed B: Time Series Analysis and Forecasting (Ballroom, II)

6:30 p.m.-7:30 p.m. Reception (Ballroom, V-VII)

7:30 p.m.-10:00 p.m. Banquet (Ballroom, III-IV)

Saturday, March 21, 1992

8:00 a.m.-10:00 a.m. Invited A: Visualization in Climate Research (Ballroom, III)
Invited B: Neural Networks (Ballroom, IV)
Invited C: High Performance Computing (Ballroom, I)
Contributed A: Small Sample Statistical Inference (Ballroom, II)

10:00 a.m.-10:30 a.m. Break (Ballroom V-VII)

10:30 a.m.-12:30 p.m. Invited A: Time Series Computing (Ballroom, III)
Invited B: Conditional Methods in Regression and Logistic Regression (Ballroom, IV)
Contributed A: Visualizing High Dimensional Data (Ballroom, I)
Contributed B: Statistical Inference (Ballroom, II)

12:30 p.m. End of Conference

TABLE OF CONTENTS

- iii Preface
- iv Cosponsoring Organizations, Cooperating Societies, and Exhibitors
- v Symposium Schedule

I. NONPARAMETRIC REGRESSION

- 1 Practical Simultaneous Nonparametric Regression Confidence Bands
R.L. Eubank, Texas A&M University, and P.L. Speckman, University of Missouri-Columbia

II. ALGORITHMS FOR MULTIVARIATE DISTRIBUTIONS AND DATA

- 9 A Novel Efficient Global Search Algorithm for Multiple Dimensions
J.F. Elder IV, University of Virginia, Charlottesville
- 18 Multivariate Stable Distributions
J.P. Nolan, American University
- 22 Self-Validating Computations of Bivariate Normal Cumulative Distribution Functions
M.C. Wang, University of Central Florida
- 25 Computation of the Multivariate Hypergeometric Distribution
T. Wu, Southern Illinois University-Edwardsville

III. SOFTWARE DESIGN AND QUALITY

- 29 Improving the Judgement of Professional Sports Managers by Using Systematic Observation of a Real-Time Process and Creating Better 'Game' Statistics (The Case of NHL Hockey)
P.A. Balthazard, University of Arizona, and K.J. Leonard, Wilfrid Laurier University
- 35 Choosing a Computational Environment for Analyzing Large Databases
P.J. Gregor, Houston Center for Quality Care and Utilization Studies
- 40 Reuse-Oriented Approach in Developing Statistical Software
Z. Lou and A. Ciampi, McGill University
- 45 Systematic Editing of Complex Questionnaire Skip Patterns
R.F. Teitel, Abt Associates

IV. VISUALIZING PROGRAMS

- 49 A Program for Identifying Duplicated Code
B.S. Baker, AT&T Bell Laboratories
- 58 Dotplot: A Program for Exploring Self-Similarity in Millions of Lines of Text and Code
K.W. Church and J.I. Helfman, AT&T Bell Laboratories
- 68 Dynamic Graphics for Software Visualization
S.G. Eick, AT&T Bell Laboratories

V. SPATIAL TIME SERIES

- 78 An Approach to Statistical Spatial-Temporal Modeling of Meteorological Fields
M.S. Handcock, New York University and J.R. Wallis, IBM T.J. Watson Research Center
- 85 Space-Time Models for the Analysis of Satellite Temperature and Ozone Data
X. Niu, Florida State University and G.C. Tiao, University of Chicago
- 95 Space-Time Modeling of Simply Connected Objects: An Application to Detection of Left Ventricular Cardiac Boundaries from Ultrasound Images
G. Storvik, Norwegian Computing Center and P. Switzer, Stanford University

VI. BUILDING ON EXISTING SOFTWARE

- 104 Tools for Managing the S Object
O. Cherkaoui and R. Ferland, University of Quebec-Montreal
- 108 A Selection of Utilities to Facilitate Data Analysis under UNIX
L. Galway and D. Relles, RAND
- 112 Design of an S Function for Robust Regression Using Iteratively Reweighted Least Squares
R.M. Heiberger, Temple University and R.A. Becker, AT&T Bell Laboratories
- 117 Statistical Computation Using GAUSS: Examples in Process Capability Research
K. Hung and D. Hagen, Western Washington University
- 121 Visualizing Experimental Designs with LISP-STAT
P.W. Iversen and M.G. Marasinghe, Iowa State University

VII. INDUSTRIAL STATISTICS

- 126 Graphical Methods for Manufacturing Yield Improvement
A.J. Black, Harris Semiconductor
- 131 A Combined Shewhart-Cumulative Score Procedure for Process Variability
M.M. Ncube, University of West Florida
- 136 Application of Experimental Design in Chemical Process Simulation
R.H. Wang and R.B. Safadi, Olin Research

VIII. VISUALIZING MULTIVARIATE DATA AND FUNCTIONS

- 141 Hierarchical Multivariate Visualization
T. Mihalisin and J. Schwegler, Temple University and Mihalisin Associates and J. Timlin, Mihalisin Associates
- 150 Graphical Methods for Permutation Data
G.L. Thompson, Southern Methodist University

IX. ADVANCED STATISTICAL TECHNIQUES FOR INDUSTRY

- 160 Methods for the Analysis of Coordinate Measurement Data
F.L. Hulting, Alcoa

- 170 Exploring Time Series Using Univariate Phase Maps

E.L. Russell III, Advanced Micro Devices

X. APPROXIMATING INTEGRALS AND DISTRIBUTIONS

- 179 Adapting Subregion-Adaptive Integration Software to Problems in Bayesian Inference

A. Erkanli, Duke University, R.E. Kass, Carnegie Mellon University, and A. Genz, Washington State University

- 182 Computing Doubly Non-Central t Probabilities

J.M. Hardin and K.P. Singh, University of Alabama-Birmingham

- 187 Asymptotic Test Statistics for Infinitely Divisible Discrete Distributions I

J.J. Hsieh, University of Toronto

XI. STOCHASTIC PROCESSES AND THEIR APPLICATIONS

- 173 Multiple Component, Linear Conditional Probability Models for Finite State Markov Processes

J.P. Benedict and W.F. Szewczyk, National Security Agency

- 198 Stochastic Epidemics on a Grid: Endemicity and Related Problems

M. Lloyd, Heriot-Watt University

- 203 On a Family of Autoregressive Processes and Cantor Sets

C. Wang, Trenton State College and A. Silverberg, American Cyanamid

XII. STATISTICAL VISUALIZATION SOFTWARE

- 208 Visualizing the Embedding of Objects in Euclidean Space

M. Littman, D.F. Swayne, N. Dean, and A. Buja, Bellcore

- 218 Statistical Visualization

J. Sall, SAS Institute

- 224 ViSta: A Visual Statistics Research and Development Testbed

F.W. Young, R.A. Faldowski, and M.M. McFarlane, University of North Carolina

XIII. WAVELETS AND NONPARAMETRIC MODELING

- 234 Advantages and Disadvantages of Density Estimation with Wavelets

G.G. Walter and J.K. Ghorai, University of Wisconsin-Milwaukee

XIV. LINEAR STATISTICAL INFERENCE

- 244 Investigating the Linearity of a Nonlinear Function

L. Denby and J.M. Landwehr, AT&T Bell Laboratories

- 249 Order and Influence in Regression Strategy

J.J. Faraway, University of Michigan

- 254 A Gibbs Sampler Approach to the Nonlinear Mixed Model

J. Gerson, Veterans Affairs Medical Center-San Francisco

- 259 Estimating Coefficients of Two-Phase Linear Regression Model with Autocorrelated Errors

T.S. Lee, Western Illinois University

263 Multiple Comparisons for Correlated Means Bounds for Conservatism

P.N. Somerville, University of Central Florida

266 Influence Diagnostics with the Gibbs Sampler

R.E. Weiss, UCLA School of Public Health

XV. APPLICATIONS OF GRAPHICS

271 Reliability Model Generator

A.J. Booker, Boeing Computer Services-Seattle

276 Computer-Aided Immunization and Opportunities for Data Visualization

G.L. Crane, New Jersey State Department of Health

283 Enhancing Tactical Direct Fire Synchronization Measures

D.A. Dryer, H. Larson, W. Kemple, and R.W. Lamont, Naval Postgraduate School

288 Exploration and Analysis of Data from a RCRA Land Treatment Facility

G.L. Patton, E.M. Armstrong, M.W. Holder, K.L. Coley, and R.C. Wallace, Radian Corporation

XVI. POSTER/VIDEO/DEMO SESSION

293 MISHA—A Computational and Graphical Tool

J.W. Hardin and H. Schmiediche, Texas A&M University

298 Interactive Analysis of Gappy Bivariate Time Series Using AGSS

P.A.W. Lewis and B.K. Ray, Naval Postgraduate School

305 A Comparative Visualization Study of a Graphical Multivariate Normality GOF Test

J.L. Romeu, SUNY-Cortland

XVII. BAYESIAN COMPUTING

310 Some Integration Strategies for Problems in Statistical Inference

M. Evans, University of Toronto and T. Swartz, Simon Fraser University

318 Posterior Integration in Dynamic Models

P. Mueller, Duke University

325 Mixture Models, Monte Carlo, Bayesian Updating, and Dynamic Models

M. West, Duke University

XVIII. STATISTICAL METHODS IN SOFTWARE QUALITY EVALUATION

334 Statistical Representations and Analyses of Software

W.M. Evancho and W.W. Agresti, The MITRE Corporation, McLean, Virginia

345 Parameter Estimation for Software Reliability Models Based on Delayed S-Shaped NHPP

A.L. Goel and K.Z. Yang, Syracuse University and R. Paul, US Army-OPTEC

350 The Role of Statistical Reliability Assessment

A. Podgurski, Case Western Reserve University

XIX. BUILDING ON S

- 359 A Graphical User Interface for S

R. Barter, N. Fisher, and M. Walmsley, CSIRO

- 366 Adding a Survival Package to S

T.M. Therneau, Mayo Foundation

XX. NONPARAMETRIC REGRESSION AND DENSITY ESTIMATION, I

- 370 The Minimal Spanning Tree for Nonparametric Regression and Structure Discovery

D. Banks, Carnegie Mellon University and M. Lavine, Duke University

- 375 On Orthogonal Series Estimators for Random Design Nonparametric Regression

S. Efromovich, University of New Mexico

- 380 A Large Scale Gaussian Smoothing Algorithm

R.K. Goodrich, University of Colorado, R.M. Passi, Institute for Naval Oceanography, and M. Limber, University of Waterloo

- 385 Computing Quantile Smoothing Splines

R.W. Koenker, University of Illinois-Urbana-Champaign and P.T. Ng, University of Houston

- 389 A New Nonparametric Estimation Method: Local and Nonlinear

A.S. Kozek, University of Wroclaw

XXI. DESIGNING AND TEACHING GRAPHICS

- 394 An Experiment to Investigate the Effects of Reduced Data Ink in Vertical Bar Charts

M. Eakin, University of Texas-Arlington

- 397 Statistical Graphics Laboratory Implementation

L.L. Hoffman, University of Central Florida

- 400 The POSTSCRIPT Imaging Model

C.C. Shing, Radford University

- 403 Handwritten Digit Recognition Using Deformable Templates

A. Sutherland, University of Strathclyde

XXII. GEOGRAPHIC INFORMATION SYSTEMS

- 408 Spatial, Statistical, and Graphical Dimensions of Data Quality

M.K. Beard, University of Maine-Orono and B.P. Battenfield, SUNY/Buffalo

- 416 Visualization of Fuzzy Scenes and Probability Fields

Y. Leung, Chinese University of Hong Kong and M.F. Goodchild and C.C. Lin, University of California-Santa Barbara

- 423 Integrating Maps and Statistical Graphs in Graphic Scripts

M. Monmonier, Syracuse University

XXIII. SAMPLING BASED APPROACHES FOR BAYESIAN INFERENCE

433 Iterative and Non-iterative Simulation Algorithms

A. Gelman, University of California-Berkeley

439 Software for the Gibbs Sampler

W.R. Gilks, A. Thomas, and D.J. Spiegelhalter, Medical Research Council Biostatistics Unit, Cambridge, UK

449 Bayesian Inference by Simulation in a Stochastic Model from Hematology

M.A. Newton, University of Wisconsin-Madison and P. Guttorp and J.L. Abkowitz, University of Washington

XXIV. UNIX TOOLS FOR STATISTICAL COMPUTING

456 Data Manipulation in Perl

D.M. Bates, University of Wisconsin-Madison

463 DE—The Data Entry Program

M. Conlon, University of Florida-Gainesville

468 Writing Unix Commands

P. Spector, University of California-Berkeley

XXV. NONPARAMETRIC REGRESSION AND DENSITY ESTIMATION, II

474 Projection Pursuit Indices Based on Fractal Dimension

J. Cabrera and D. Cook, Rutgers University

478 An Analysis of Polynomial-Based Projection Pursuit

D. Cook, Bellcore and Rutgers University, A. Buja, Bellcore, and J. Cabrera, Rutgers University

483 Maximum Entropy Density Estimation using Random Tessellations

L.B. Hearne and E.J. Wegman, George Mason University

488 An Efficiency Study in Nonparametric Regression with Correlated Errors

D.B. Holiday, University of Texas Health Center at Tyler

493 A Hierarchical Genetic System for Symbolic Function Identification

M. Jiang and A.H. Wright, University of Montana

XXVI. TIME SERIES ANALYSIS AND FORECASTING

498 A Study of Neural Network Design and its Application to Statistical Forecasting

Y.R. Park and C. Chen, Syracuse University and T.J. Murray, New Jersey Institute of Technology

XXVII. HIGH PERFORMANCE COMPUTING

503 Smoothing Census Adjustment Factors: An Application of High Performance Computing

W.F. Eddy, M.M. Meyer, A. Mockus, M.J. Schervish, K. Tan, and K. Viele, Carnegie Mellon University

XXVIII. SMALL SAMPLE STATISTICAL INFERENCE

- 511 Bootstrap Confidence Interval Estimation for Model-Based Surveys

K. Do, Australian National University and P. Kokic, Australian Bureau of Agriculture and resource Economics

- 516 Graphic Representation of the Effects of Skewness and Kurtosis on the Power of the Two-Sided Student's t Test

M. Eakin, University of Texas-Arlington and S. Bowman, Alcon Laboratories

- 519 Power Comparisons for Two-Sample Rank Tests

S. Jain and J.G.C. Templeton, University of Toronto

- 524 Small Sample Empirical Critical Values as a Tool for the Comparison of Multivariate Normality GOF Tests

J.L. Romeu, SUNY-Cortland

- 529 On the Accuracy of Binomial and Proportion Estimators: an Absolute Rule

E.F. Schuster, University of Texas-El Paso

XXIX. TIME SERIES COMPUTING

- 534 Simulating Gaussian Random Processes with Specified Spectra

D.B. Percival, University of Washington

- 539 Fast Algorithms for Modeling Rapidly Sampled Data

H.V. Poor, Princeton University

XXX. CONDITIONAL METHODS IN REGRESSION AND LOGISTIC REGRESSION

- 546 Resampling Permutations in Regression with Exchangeable Errors

R. LePage and K. Podgorski, Michigan State University

- 554 Exact Logistic Regression: Theory, Applications, and Software

C.R. Mehta and N.R. Patel, Harvard University and Cytel Software Corporation

XXXI. VISUALIZING HIGH DIMENSIONAL DATA

- 565 Visualizing Correlation Decision Making in Data Fusion Problems

D.E. Brown, J.F. Elder IV, and C.L. Pittard, Jr., University of Virginia, Charlottesville

- 569 Assisted Visual Search for Data Structure

J.F. Elder IV, University of Virginia, Charlottesville

- 575 Classification Procedures and its Application

R.K. Jain, Memorial University of Newfoundland

- 578 The Application of Kriging for the Controlled Minimization of Large Data Sets

D. Robinson and C. Brodtkin, Air Force Institute of Technology

- 582 Visualizing Cluster Structure in High Dimensions

E. Tabakis, Massachusetts Institute of Technology

XXXII. STATISTICAL INFERENCE

- 587 Graphical and Formal Tests of Fit for Discrete Distributions based on Probability Generating Functions

T.W. Epps, University of Virginia, Charlottesville

- 592 Optimal Allocation for Estimating the Product of Two Means

J.P. Hardwick and Q.F. Stout, University of Michigan

- 597 Optimal Adaptive Equal Allocation Rules

J.P. Hardwick and Q.F. Stout, University of Michigan

XXXIII. BANQUET ADDRESS

- 602 History of Statistics in Real Time: Hammers and Nails

E. Parzen, Texas A&M University

Practical Simultaneous Nonparametric Regression Confidence Bands

R. L. Eubank*
Department of Statistics
Texas A&M University

P. L. Speckman
Department of Statistics
University of Missouri-Columbia

Abstract

Two proposals for simultaneous confidence bands in nonparametric regression are examined, one based on bias-correction and a second using a Bonferroni inequality. Both are implemented with data-driven bandwidth selection and nonparametric variance estimation. Asymptotic theory shows both to have suitable coverage for large samples, and simulation suggests good coverage properties for samples as small as 50. An application of the methodology is used to give bounds for coverage probabilities in nonparametric regression with samples ranging from 50 to 1000.

1. Introduction

This article reports on progress in confidence bands associated with nonparametric regression estimation. There has been a great deal of development of the analytical properties of nonparametric estimators, but the corresponding technology for confidence bands has lagged. In this paper we outline theoretical and experimental results aimed at providing users with practical bands.

Consider the situation where responses y_1, \dots, y_n are obtained at equally spaced design points $t_r = r/n$, $r = 1, \dots, n$. The y_r and t_r are related under the model

$$y_r = \mu(t_r) + \varepsilon_r, r = 1, \dots, n, \quad (1.1)$$

where the ε_r are independent, identically distributed, random variables having zero means and common variance σ^2 , and μ is an unknown, smooth regression curve. The problem to be addressed is the construction of confidence bands for μ . Specifically, given $\alpha \in (0, 1)$ and an estimator $\hat{\mu}$ for μ , we want to find data-based bounds $L_\alpha(t)$ and $U_\alpha(t)$, presumably depending on $\hat{\mu}(t)$, such that

$$P(L_\alpha(t) \leq \mu(t) \leq U_\alpha(t) \text{ for all } t) \approx 1 - \alpha, \quad (1.2)$$

at least in large samples. To be practical, these bounds (1) should be completely data-driven and (2) should provide a reasonably good (or at least conservative) approximation to the nominal confidence level.

*Research supported in part by NSF Grant DMS-9024879

To illustrate the problem of nonparametric bands in a specific setting, consider the case of a second-order kernel estimator for μ of the form

$$\mu_\lambda(t) = \frac{1}{n\lambda} \sum_{r=1}^n y_r K\left(\frac{t-t_r}{\lambda}\right),$$

where $\lambda > 0$ is the bandwidth and K is a symmetric kernel supported on $[-1, 1]$. If $\mu(t)$ possesses two continuous derivatives and K is a second order kernel, it is well known that

$$E(\mu_\lambda(t)) - \mu(t) \approx \frac{\lambda^2 \mu''(t)}{2} \int_{-1}^1 x^2 K(x) dx \quad (1.3)$$

$$\begin{aligned} \text{Var}(\mu_\lambda(t)) &\approx \frac{\sigma^2}{n\lambda} \int_{-1}^1 K(x)^2 dx \\ &= \sigma^2 V_n(\lambda). \end{aligned} \quad (1.4)$$

Consequently, if the usual smoothing criterion of minimizing

$$\text{AMSE}(\lambda) = \frac{1}{n} \sum_{r=1}^n E(\mu_\lambda(t_r) - \mu(t_r))^2$$

is adopted, the optimal bandwidth λ_0 will satisfy

$$\lambda_0 \sim kn^{-1/5} \quad (1.5)$$

where k is a constant depending only on K and μ . For clarity of exposition, the rest of this article concentrates solely on this setting of second order asymptotics with the related rates of convergence. While we have not explicitly worked out the details, these results presumably extend to higher order kernels.

Typical large sample theory for confidence bands is based on the approximate normality of a pivot statistic like $T_\lambda = [\mu_\lambda(t) - \mu(t)]/\sigma\sqrt{V_n(\lambda)}$. But at the optimal smoothing parameter λ_0 , the bias of μ_λ is not negligible. Writing

$$T_\lambda = \frac{\mu_\lambda(t) - E(\mu_\lambda(t))}{\sigma\sqrt{V_n(\lambda)}} + \frac{E(\mu_\lambda(t)) - \mu(t)}{\sigma\sqrt{V_n(\lambda)}}, \quad (1.6)$$

the first term, the stochastic portion, is asymptotically standard normal under weak conditions, but the second

term is in general $O(1)$. To use T_λ as a pivot statistic directly, we need to choose λ in such a way that the second term goes to zero. Data-driven methods for estimating an optimal λ , generally designed to obtain a rate like (1.5), will not achieve this goal. Thus the dilemma of nonparametric confidence bounds has been either to deliberately undersmooth, thereby employing an inferior estimate of $\mu(t)$, or to compensate for the unknown bias term in (1.6) explicitly, thereby changing the presumed otherwise optimal properties of the estimator.

The first approach, that of deliberately undersmoothing, has been taken by a number of authors beginning perhaps with Bickel and Rosenblatt (1973) in their study of kernel density estimation. Extensions to nonparametric regression include results by Johnston (1982) and Härdle (1989).

Another approach to the bias problem is to incorporate an *a priori* bound on the bias in a confidence band. Knafl, Sacks, and Ylvisaker (1982) showed that it is possible to obtain conservative bounds on the simultaneous coverage probability in this fashion. Hall and Titterton (1988) in related work obtained best possible simultaneous confidence band lengths under similar assumptions. They showed that the minimum width of uniform confidence bands of the form (1.2) is

$$\left(\frac{\log n}{n}\right)^{2/5}. \quad (1.7)$$

Both of these methods require additional information and hence are not completely data-driven.

Direct approaches for dealing with the bias term have also been used. (Hall (1990) used the term *explicit bias estimation* to denote such confidence bands.) For example, Cox (1986) used an estimate of the the norm of the bias in constructing simultaneous confidence bands for smoothing splines. More recently, Härdle and Bowman (1988) and Härdle and Marron (1991) introduced an explicit bias estimate as part of a bootstrap method for estimating the distribution of $\sup_t |T_\lambda(t)|$ itself rather than relying on asymptotic theory. However, the latter approach requires selection of an auxiliary bandwidth for bias correction in the bootstrap process, a topic apparently not fully addressed in the literature.

A third alternative is the use of Bonferroni bounds. Härdle (1990) reports on an application of this idea, but to our knowledge it has not been explored further.

In the rest of this paper, we examine two methods, one based on bias-correction and the other a direct application of a Bonferroni inequality. We show that it is possible, asymptotically, to formulate completely data-driven methods that provide at least guaranteed asymptotic simultaneous coverage, and we provide Monte Carlo

evidence that the methods are practical with samples as small as 50.

2. Bias-corrected bands

In order to have a tractable theory, we make several simplifying assumptions (c.f. Rice (1984)). For the remainder of this article, we assume that $\mu(t)$ is periodic on $(0,1)$ and has two continuous periodic derivatives satisfying

$$\mu^j(0) = \mu^j(1), \quad j = 0, 1, 2. \quad (2.1)$$

2.1. Theoretical background

Much of the theoretical development for nonparametric confidence bands is based on the seminal work of Bickel and Rosenblatt (1973). The following version of one of their results holds in the context of fixed-design nonparametric regression.

Lemma 1 *Let*

$$Z_\lambda(t) = \frac{\mu_\lambda(t) - E(\mu_\lambda(t))}{\sigma \sqrt{V_n(\lambda)}}$$

and

$$M_\lambda = \sup_{0 \leq t \leq 1} |Z_\lambda(t)|.$$

If K is twice continuously differentiable and ϵ_r has four absolute moments, then

$$P\left(M_{\lambda_0} \leq \sqrt{-2 \log \lambda_0} + \frac{C + x_\alpha}{\sqrt{-2 \log \lambda_0}}\right) \rightarrow 1 - \alpha,$$

where

$$x_\alpha = -\log |\log(1 - \alpha)|$$

$$C = \log \left(\frac{1}{2\pi} \left[\int_{-1}^1 K'(u)^2 du \right] / \left[\int_{-1}^1 K(u)^2 du \right] \right)^{1/2}.$$

(NB the Bickel and Rosenblatt (1975) correction to the constant C .) This result is proven using a strong approximation argument in Eubank and Speckman (1992).

Define

$$\ell_\alpha^* = \sigma \sqrt{V_n(\lambda_0)} \left(\sqrt{-2 \log \lambda_0} + \frac{C + x_\alpha}{\sqrt{-2 \log \lambda_0}} \right). \quad (2.2)$$

A direct consequence of Lemma 1 is that the interval

$$\left\{ \mu : \sup_{0 \leq t \leq 1} |\mu_{\lambda_0}(t) - \mu(t)| \leq \ell_\alpha^* \right\} \quad (2.3)$$

has asymptotic coverage probability $1 - \alpha$ for $E(\mu_{\lambda_0})$. In the absence of bias, or when the bias is asymptotically negligible (for example, as when $\lambda = o(n^{-1/5})$),

this constitutes a valid method. (See Johnston (1982) and Härdle (1989), e.g., for this approach in the random design nonparametric regression problem.) With the more usual $O(n^{-1/5})$ bandwidth commonly recommended for practical use, an obvious way to attempt to convert (2.3) into a confidence band for μ is to estimate the bias, $E(\mu_{\lambda_0}(t) - \mu(t))$, directly, and to incorporate the estimate into the interval.

Motivated by (1.3), we take

$$b_{\lambda}(t) = \lambda^2 B \hat{\mu}''(t)$$

as an estimate of the bias, where B is a known constant and $\hat{\mu}''(t)$ is a suitable estimator of $\mu''(t)$. One possibility is

$$\hat{\mu}''(t) = \frac{1}{n \bar{\lambda}^3} \sum_{r=1}^n y_r K^* \left(\frac{t - t_r}{\bar{\lambda}} \right), \quad (2.4)$$

where K^* is a square integrable kernel supported on $[-1, 1]$ satisfying $\int_{-1}^1 u^j K^*(u) du = 0$, $j = 0, 1$, and $\int_{-1}^1 u^2 K^*(u) du = 2$, and $\bar{\lambda}$ is a bandwidth converging to zero at the rate $n^{-1/7}$ (c.f. Müller (1988)).

Under suitable conditions, $\mu_{\lambda}(t) - b_{\lambda}(t)$ should have less bias than $\mu_{\lambda}(t)$ itself. Surprisingly, the variance of the corrected estimator is not seriously altered. In fact, any estimator of the type (2.4) is good enough in the following sense, as shown in Eubank and Speckman (1992).

Lemma 2 *If μ'' is Lipschitz continuous of order $\nu > 0$ and K^* is twice continuously differentiable, then*

$$\sup_{0 \leq t \leq 1} \frac{|E(\mu_{\lambda_0}(t)) - \mu_{\lambda_0}(t) - b_{\lambda_0}(t)|}{\sigma \sqrt{V_n(\lambda_0)}} = O_p(n^{-\theta})$$

for some $\theta > 0$.

This result combined with Lemma 1 immediately shows that the bias-corrected confidence band

$$\left\{ \mu : \sup_{0 \leq t \leq 1} |\mu_{\lambda_0}(t) - b_{\lambda_0}(t) - \mu(t)| \leq \ell_{\alpha}^* \right\}$$

has the correct asymptotic simultaneous coverage probability.

To be practical, this method requires choices or estimates for λ_0 , $\bar{\lambda}$, and σ . In what follows, we require that $\bar{\lambda}$ be any estimator of λ_0 with the property

$$\frac{\bar{\lambda} - \lambda_0}{\lambda_0} = O_p(n^{-1/10}). \quad (2.5)$$

Rice (1984) and Härdle, Hall and Marron (1988) have shown that (2.5) is satisfied by many bandwidth estimators including the one obtained from generalized cross-validation. The auxiliary bandwidth $\bar{\lambda}$ for estimating μ''

may not be as critical. We take $\bar{\lambda} = \hat{\lambda}^{5/7}$, a data-driven bandwidth with the requisite $n^{-1/7}$ rate. Finally, we typically must estimate σ . Any \sqrt{n} -consistent estimator can be used such as one of those found in Gasser, Sroka and Jennen-Steinmetz (1985), Hall, Kay and Titterton (1990) or Hall and Marron (1990). Under these conditions, let

$$\ell_{\alpha} = \hat{\sigma} \sqrt{V_n(\hat{\lambda})} \left(\sqrt{-2 \log \hat{\lambda}} + \frac{C + x_{\alpha}}{\sqrt{-2 \log \hat{\lambda}}} \right). \quad (2.6)$$

Then the following result is proven in Eubank and Speckman (1992).

Theorem 1 *Assume (2.1) and (2.5) hold and that the ϵ 's possess more than nineteen absolute moments. Under the conditions of Lemmas 1 and 2,*

$$P \left(\sup_{0 \leq t \leq 1} |\mu_{\bar{\lambda}}(t) - b_{\bar{\lambda}}(t) - \mu(t)| \leq \ell_{\alpha} \right) \rightarrow 1 - \alpha. \quad (2.7)$$

2.2. Practical considerations

Our experience suggests that obtaining accurate coverage rates is a delicate matter. One clue to the problem may be that the effect of α on the width of the band ℓ_{α} is of second order. It's clear from (2.2) that $\ell_{\alpha}^*/\ell_{\alpha'}^* \rightarrow 1$ as $n \rightarrow \infty$ for any fixed $0 < \alpha, \alpha' < 1$. This observation, characteristic of extreme value distributions, points out a practical problem in obtaining good coverage probabilities. Seemingly minor adjustments to the algorithm can have a large impact not suggested by the theory, especially for small samples or large σ .

One significant factor is the correct evaluation of the variance term $V_n(\lambda)$. According to Lemma 2, the bias adjustment $b_{\lambda}(t)$ is uniformly negligible, and the result of the Theorem follows in part by an application of Slutsky's theorem. However, for finite samples, the added variability from estimating the bias can have substantial impact on coverage probability. In our current implementation, we now compute $V_n(\lambda)$ as a sum involving the squared coefficients of the y_r 's in $\mu_{\lambda}(t) - b_{\lambda}(t)$.

As mentioned above, we need an auxiliary bandwidth for estimating $\mu''(t)$. Among several possibilities we tried, the choice $\bar{\lambda} = \hat{\lambda}^{5/7}$ seems to work well. There may be better choices, and the algorithm could undoubtedly be further tuned.

It is interesting to compare the width of confidence band (2.7) with the theoretical best possible rate (1.7). From (2.6) and the fact that $\hat{\lambda} = O_p(n^{-1/5})$, it is evident that

$$\ell_{\alpha} \sim cn^{-2/5} \sqrt{\log n} + \text{lower order terms},$$

giving a rate very close to the theoretical optimum.

3. Bonferroni bands

The Bonferroni confidence band, constructed by using a simple Bonferroni inequality to bound the maximum stochastic deviation in (1.6) and ignoring bias, has the form

$$\left\{ \mu : \sup_{0 \leq t \leq 1} |\mu_{\lambda_0}(t) - \mu(t)| \leq \ell_{B\alpha} \right\}, \quad (3.1)$$

with

$$\ell_{B\alpha} = z_{\alpha/2n} \hat{\sigma} \sqrt{V(\hat{\lambda})}$$

and $z_{\alpha/2n}$ the $100(1 - \alpha/2n)$ th percentile of the standard normal distribution. To see why (3.1) works, note that

$$P \left(\max_{1 \leq r \leq n} |\mu_{\hat{\lambda}}(t_r) - \mu(t_r)| \leq \ell_{B\alpha} \right)$$

is at least as large as

$$P \left(\sup_{0 \leq t \leq 1} |\mu_{\hat{\lambda}}(t) - b_{\hat{\lambda}}(t) - \mu(t)| \leq \ell_{B\alpha} - \sup_{0 \leq t \leq 1} |b_{\hat{\lambda}}(t)| \right).$$

The Bonferroni bands will have asymptotic coverage at least $1 - \alpha$ if

$$P \left(\left[\ell_{B\alpha} - \sup_{0 \leq t \leq 1} |b_{\hat{\lambda}}(t)| \right] / \ell_{\alpha} > 1 \right) \rightarrow 1.$$

But this is true because

$$\ell_{B\alpha} \sim \sigma \sqrt{V_n(\hat{\lambda})} \sqrt{2 \log n} = O_p(n^{-2/5} \sqrt{\log n}),$$

using the standard extreme value approximation for the standard normal distribution, and

$$\sup_{0 \leq t \leq 1} |b_{\hat{\lambda}}(t)| = O_p(\hat{\lambda}^2) = O_p(n^{-2/5}),$$

while ℓ_{α} behaves like $\sigma \sqrt{V_n(\hat{\lambda})} \sqrt{(2/5) \log n}$.

Note that the foregoing discussion implies that

$$\ell_{B\alpha} / \ell_{\alpha} \rightarrow_p \sqrt{5} \approx 2.24.$$

This suggests that Bonferroni bands, while asymptotically considerably larger than the bounds employing the Bickel-Rosenblatt asymptotic distribution, at least have the same first-order rate of convergence to zero. This is somewhat comforting. A curious consequence is that $\ell_{B\alpha} / \ell_{\alpha} \rightarrow_p \sqrt{5}$ as $n \rightarrow \infty$ for any $0 < \alpha, \alpha' < 1$. However, as seen in simulation evidence below, for finite samples, the performance of the Bonferroni band may not be as bad as asymptotic theory predicts. In fact, Bonferroni bands can be narrower than bias-corrected ones.

n	α	truth	no bias-corr.	bias-corr.	Bonf.
50	0.20	0.8106	0.6264	0.9112	0.8852
50	0.10	0.9136	0.7732	0.9682	0.9292
50	0.05	0.9680	0.8724	0.9890	0.9582
50	0.01	0.9970	0.9690	0.9996	0.9826
100	0.20	0.8012	0.5928	0.8192	0.8418
100	0.10	0.9196	0.7730	0.9174	0.8964
100	0.05	0.9676	0.8694	0.9672	0.9342
100	0.01	0.9960	0.9742	0.9966	0.9722
200	0.20	0.7958	0.6018	0.7798	0.8690
200	0.10	0.9118	0.7690	0.9052	0.9082
200	0.05	0.9674	0.8756	0.9610	0.9384
200	0.01	0.9974	0.9706	0.9956	0.9742
300	0.20	0.7982	0.6052	0.7796	0.8932
300	0.10	0.9118	0.7768	0.9010	0.9338
300	0.05	0.9662	0.8818	0.9624	0.9580
300	0.01	0.9954	0.9786	0.9952	0.9838

Table 1: Empirical coverage probabilities with $\sigma = .10$.

4. Finite sample properties

To assess the performance of the two methods, we have conducted a number of Monte Carlo experiments. In one example, data were generated from model (1.1) using normal errors and the regression curve

$$\mu(t) = e^{-32(t-.5)^2}. \quad (4.1)$$

Sample sizes $n = 50, 100, 200, 300$ were employed, and σ was chosen to be .10 and .20. Confidence bands were then investigated for $\alpha = .01, .05, .10$ and .20.

A plot for a typical data set from the simulation for $n = 100$ and $\sigma = .10$ is shown in Figure 1 along with the true regression function and a kernel estimator with bandwidth selected by generalized cross-validation. Figure 2 shows the bias-corrected and Bonferroni bounds with $\alpha = .10$.

For each experimental setting we generated 5000 replicate samples. Kernel estimates were fit to the data using the Epanechnikov kernel $K(u) = .75(1 - u^2)$, $-1 \leq u \leq 1$, with λ selected by generalized cross-validation. The Hall *et al* (1990) estimator of σ was used. The proportion of times that all the $\mu(t_r)$, $r = 1, \dots, n$, fell inside the bias-corrected and Bonferroni bands was recorded as well as the average (over replications) band half-lengths ℓ_{α} and $\ell_{B\alpha}$. Standard errors for the half-lengths over the replications and the average of the estimates of σ were also recorded. In addition, coverage

probabilities were computed for two other intervals,

$$\mu_{\hat{\lambda}}(t) - \frac{1}{n\hat{\lambda}} \sum_{r=1}^n \mu(t_r) K\left(\frac{t-t_r}{\hat{\lambda}}\right) \pm \ell_{\alpha}^*$$

and

$$\mu_{\hat{\lambda}}(t) \pm \ell_{\alpha}^*.$$

These represent respectively coverage with no bias when σ is known ("truth") and coverage when σ is known but bias is ignored ("no bias-correction"). The results of the simulation are presented in Tables 1 and 2. Note that the empirical coverage probabilities and interval half-lengths for $\alpha = .01, .05, .10$ and $.20$ were computed on the same set of replications for each sample size. Thus the results for each sample size are not independent (and the average of the estimates of σ are identical).

n	α	bias-corrected	Bonferroni	$\hat{\sigma}$
50	0.20	0.1252 (.0190)	0.1263 (.0167)	0.1157
50	0.10	0.1415 (.0202)	0.1356 (.0179)	0.1157
50	0.05	0.1571 (.0215)	0.1444 (.0191)	0.1157
50	0.01	0.1924 (.0243)	0.1632 (.0216)	0.1157
100	0.20	0.0850 (.0107)	0.0912 (.0101)	0.1037
100	0.10	0.0956 (.0114)	0.0971 (.0107)	0.1037
100	0.05	0.1058 (.0121)	0.1027 (.0113)	0.1037
100	0.01	0.1288 (.0137)	0.1148 (.0127)	0.1037
200	0.20	0.0639 (.0079)	0.0720 (.0077)	0.1007
200	0.10	0.0714 (.0084)	0.0762 (.0082)	0.1007
200	0.05	0.0787 (.0089)	0.0802 (.0086)	0.1007
200	0.01	0.0951 (.0099)	0.0888 (.0095)	0.1007
300	0.20	0.0546 (.0065)	0.0633 (.0065)	0.1002
300	0.10	0.0609 (.0068)	0.0667 (.0068)	0.1002
300	0.05	0.0670 (.0072)	0.0700 (.0072)	0.1002
300	0.01	0.0807 (.0080)	0.0772 (.0079)	0.1002

Table 2: Average confidence band half-lengths (standard deviations) and average estimated σ .

Several conclusions can be drawn from this experiment. First, the Bickel-Rosenblatt approximation for the maximum random error ("truth") is quite good, even when λ is chosen from the sample. This observation remained true for other combinations of α and σ not reported here. Second, the "no bias-correction" case demonstrates that bias in μ_{λ} is a serious problem and leads to substantial undercoverage. However the bias correction technique seems to be effective. Finally, the Bonferroni method is conservative (or nearly so) in all cases despite ignoring bias.

Table 2 demonstrates that Bonferroni bands can be narrower than bias-corrected ones, especially for small α . Surprisingly, this can be observed even for sample

sizes as large as 300. Table 2 also demonstrates the bias in the Hall *et al* (1990) estimator of σ . This bias may well work to negate the effect of ignoring the bias in μ_{λ} for the Bonferroni estimator.

5. Application to a statistical experiment

Monte Carlo simulation is a powerful and frequently used tool in the study of new statistical methodology. Simulations are used to assess the performance of a method whose properties would be otherwise intractable or to verify asymptotic calculations. Characteristics frequently studied such as type I error rates, power, and coverage probabilities can often be thought of as depending smoothly on factors such as sample size or unknown parameters. These smooth relationships are rarely directly computable, hence the investigator resorts to a simulation study. Because nonparametric regression techniques are designed to estimate smooth curves, tools from the area seem ideally suited to the interpretation of many kinds of simulation experiments. This approach is related to the Bayesian analysis of the outcome of computer experiments as outlined in Sacks, Welch, Mitchell and Wynn (1989). In the context of this paper, simulations provide an ideal application for nonparametric regression methodology. Here is one case where it may be possible to generate the very large samples required by nonparametric regression.

As a concrete example, consider the problem of estimating coverage probability of bias-corrected confidence bands under model (1.1) with mean function $\mu(t)$ given by (4.1), $\sigma = .10$, and $\alpha = .10$. It seems intuitive that coverage probability is a smooth function of sample size. How can this information be used to improve the Monte Carlo estimate of coverage probability and to give confidence bounds on the estimate?

There are a number of interesting design questions for this simulation experiment. From preliminary work, we felt that the relationship between coverage probability and sample size could best be modeled on a log scale in sample size. One hundred sample sizes n_1, \dots, n_{100} were chosen between 50 and 1000 with $n_1 = 50$ and $n_{i+1}/n_i \approx \text{constant}$ subject to the restriction that sample sizes must be integer. A total of 1,600 replications at each sample size were made (for a total of 160,000 replications). The proportion of times all the $\mu(t_r)$ fell within the confidence bands for all t_r was recorded at each sample size. Thus the output from the simulation consisted of 100 data points. Assuming that the coverage level is approximately constant over sample size, the sample proportions are approximately normally distributed with constant variance. Using the log scale for

sample size meant that most of the computing effort was expended for relatively fast cases, so the project was feasible on a network of four SPARC2 processors.

The raw data from the simulation for the bias-corrected method with $\sigma = .10$ and $\alpha = 10$ are shown in Figure 3. The fitted curve estimating coverage probability was calculated by Gasser-Müller kernel smoothing using the Epachnikov kernel on the interior, boundary kernels at the end points, and bandwidth chosen by cross-validation. The smooth was computed on the log scale. To illustrate the kind of application possible with simultaneous confidence bands, Figure 4 shows a 95% confidence band computed by the boundary-correction method for the data in Figure 3. Because the standard errors were computed from the terms of the boundary kernels used for μ_λ and $\hat{\mu}''$, the confidence band widens at the boundaries.

This simultaneous confidence band is not fully supported by the results of section 2 because the design is not exactly equally spaced and because true coverage probability is not a periodic function of sample size. We conjecture that the results of section 2 extend to this situation. At worst, the results can easily be extended to all of the interval except near the endpoints. In Figure 4, the boundary regions extend from 0 to approximately 70 and from approximately 700 to 1000. Thus the confidence band appears valid at least over the range of sample sizes between 70 and 700. Note that from approximately 140 on, the band contains the nominal 90% coverage probability, and the width is only about 1% over much of the region. We believe that this graph with its confidence band gives helpful information on the uncertainty of the estimated coverage probabilities.

6. References

- Bickel, P. J. and Rosenblatt, M. (1973), "On Some Global Measures of the Deviations of Density Function Estimates," *Annals of Statistics*, 1, 1071-1095.
- Cox, D. (1986), "Approximation theory of method of regularization estimators: applications," in *Contemporary Mathematics Volume 59: Function Estimates*, J. S. Marron, ed., American Mathematical Society: Providence, 105-124.
- Eubank, R. and Speckman, P. (1992), "Confidence bands in nonparametric regression," submitted.
- Gasser, T., Sroka, L. and Jennen-Steinmetz, C. (1986), "Residual Variance and Residual Pattern in Nonlinear Regression," *Biometrika*, 73, 625-633.
- Hall, P. (1990), "Bootstrap confidence intervals in curve estimation," *Proceedings of the 22nd Symposium on the Interface*, C. Page and R. LePage, ed., Springer-Verlag: New York, 1-15.
- Hall, P., Kay, J. W. and Titterton, D. M. (1990), "Asymptotically Optimal Difference Based Estimation of Variance in Nonparametric Regression," *Biometrika*, 77, 521-528.
- Hall, P. and Marron, S. (1990), "On Variance Estimation in Nonparametric Regression," *Biometrika*, 77, 415-419.
- Hall, P. and Titterton, D. M. (1988), "On Confidence Bands in Nonparametric Density Estimation and Regression," *Journal of Multivariate Analysis*, 27, 228-254.
- Härdle, W. (1989), "Asymptotic Maximal Deviation of M-Smoothers," *Journal of Multivariate Analysis*, 29, 163-179.
- Härdle, W. (1990), *Applied Nonparametric Regression*, Cambridge University Press: Cambridge.
- Härdle, W. and Bowman, A. W. (1988), "Bootstrapping in Nonparametric Regression: Local Adaptive Smoothing and Confidence Bands," *Journal of the American Statistical Association*, 83, 102-110.
- Härdle, W., Hall, P. and Marron, J. S. (1988), "How Far Are Automatically Chosen Regression Smoothing Parameters From Their Optimum?" (with discussion), *Journal of the American Statistical Association*, 83, 86-99.
- Härdle, W. and Marron, S. (1991), "Bootstrap Simultaneous Error Bars for Nonparametric Regression," *Annals of Statistics*, 19, 778-796.
- Knafl, G., Sacks, J. and Ylvisaker, D. (1982), "Model Robust Confidence Intervals," *Journal of Statistical Planning and Inference*, 6, 319-334.
- Johnston, G. J. (1982), "Probabilities of Maximal Deviations for Nonparametric Regression Function Estimates," *Journal of Multivariate Analysis*, 12, 402-414.
- Müller, H. G. (1988), *Nonparametric Regression Analysis of Longitudinal Data*, Springer-Verlag: New York.
- Rice, J. (1984), "Bandwidth Choice for Nonparametric Regression," *Annals of Statistics*, 12, 1215-1230.
- Sacks, J., Welch, W., Mitchell, T. and Wynn, H. (1989), "Design and analysis of computer experiments (with discussion)," *Statistical Science*, 4, 409-435.

Figure 1. Data with regression curve and kernel smooth.

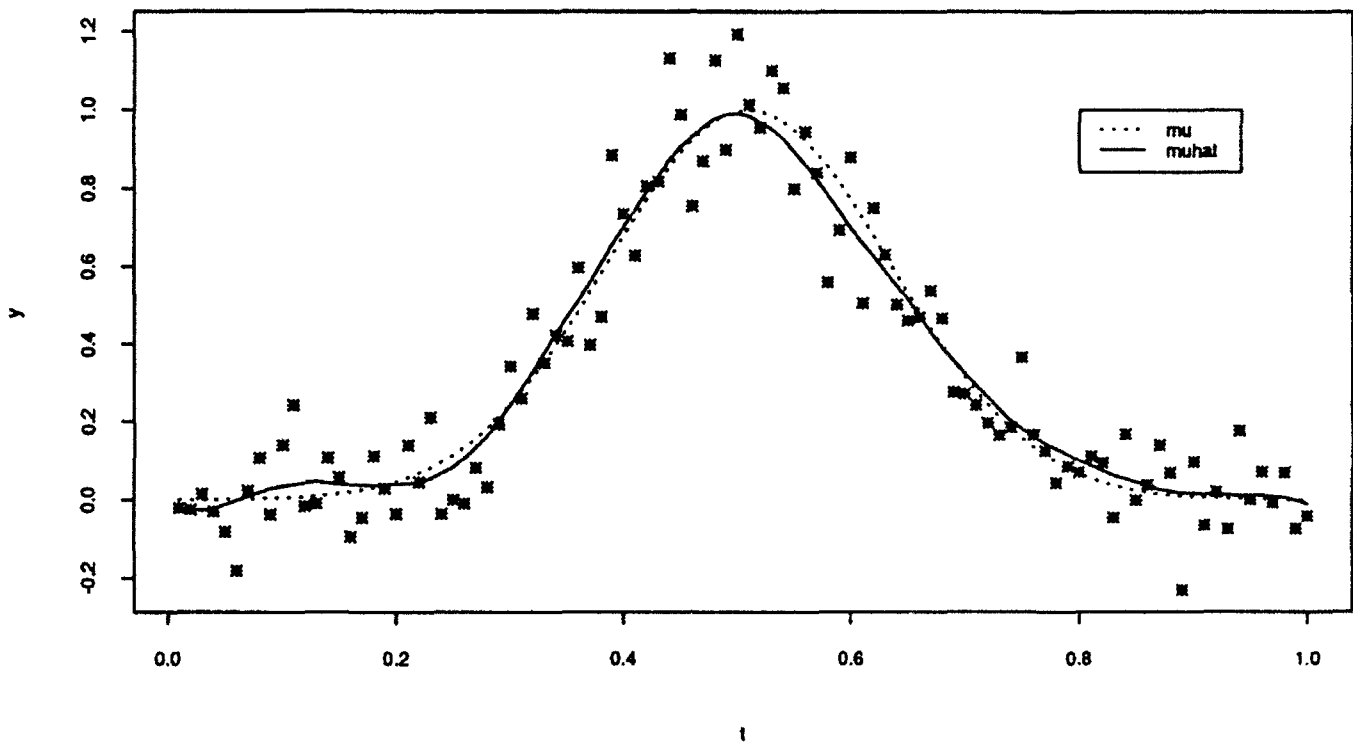


Figure 2. 90% bias-corrected and Bonferroni bands for data in Figure 1.

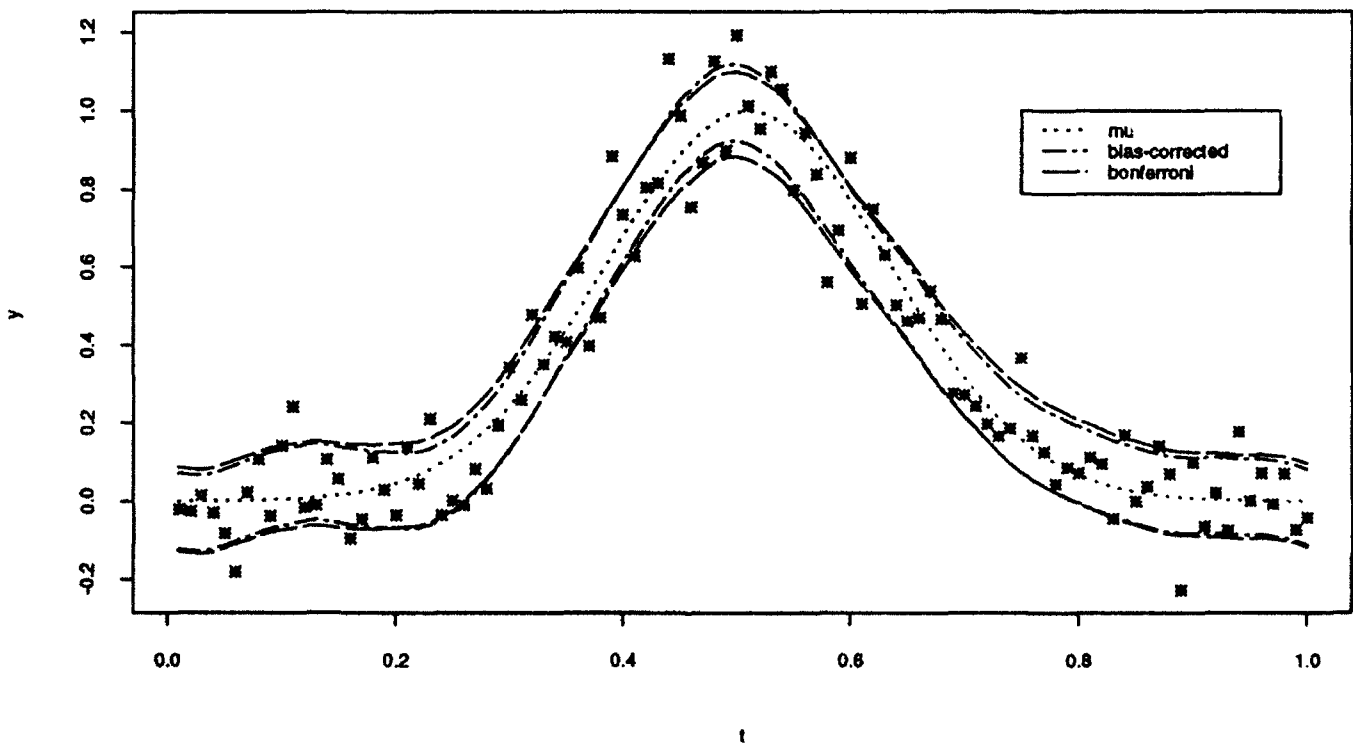


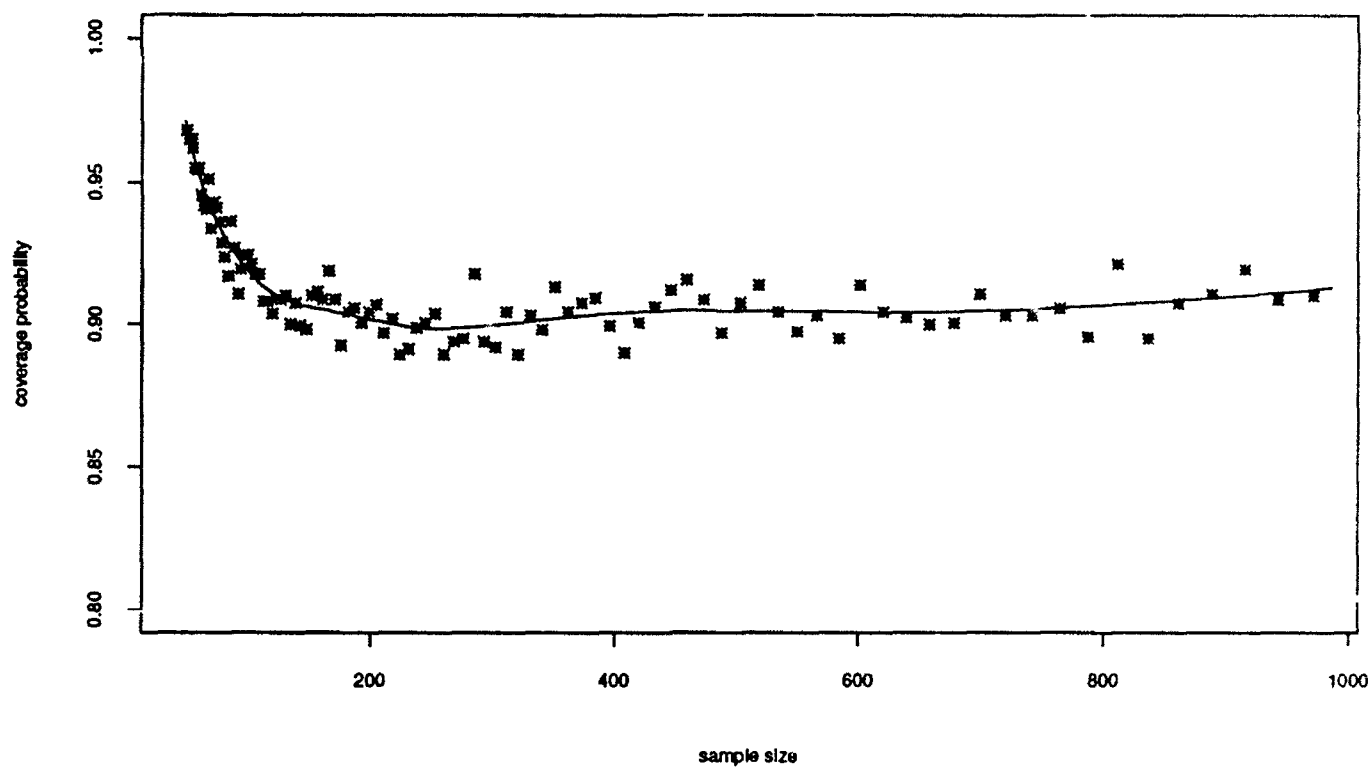
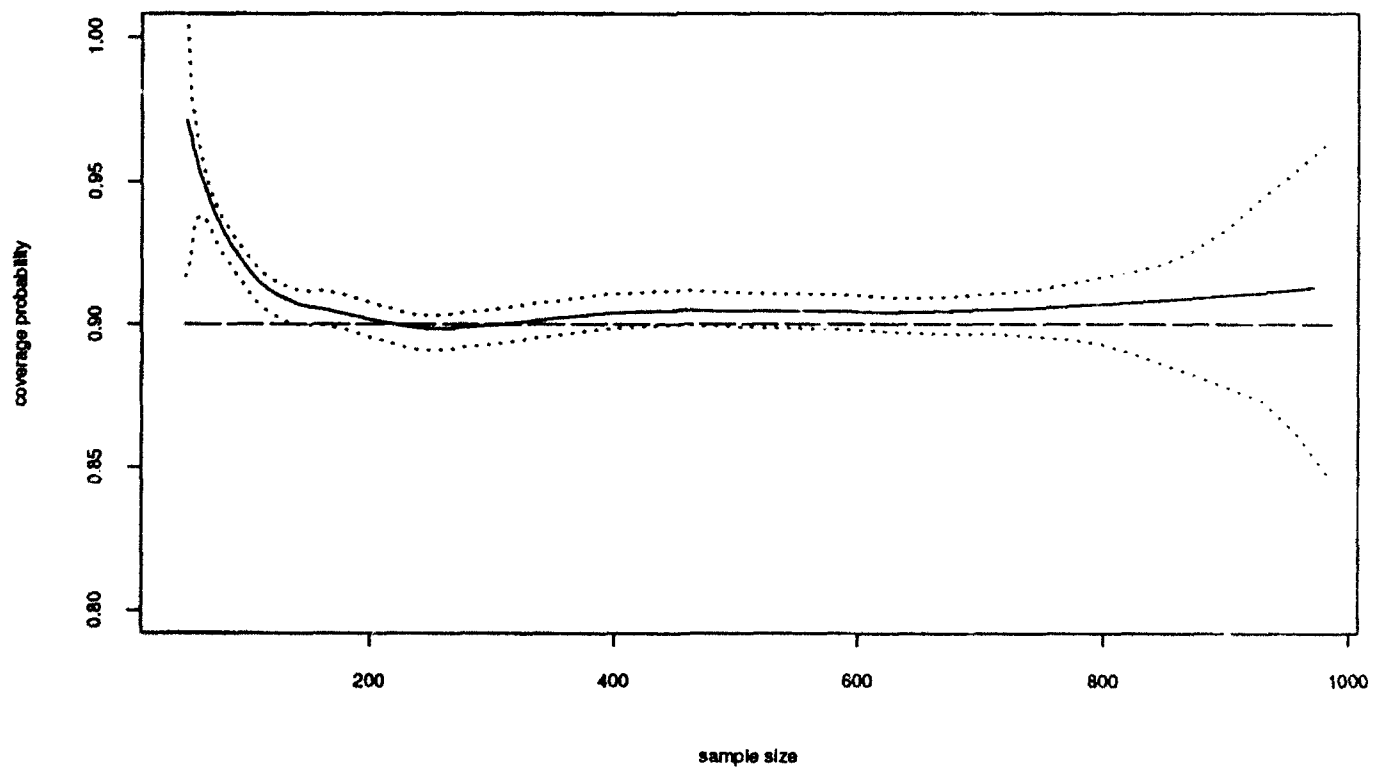
Figure 3. Empirical coverage probabilities for $\alpha = .10$ with fitted curve.

Figure 4. 95% simultaneous confidence band for coverage probability data of Figure 3.



A Novel Efficient Global Search Algorithm for Multiple Dimensions

John F. Elder IV
 Department of Systems Engineering
 University of Virginia
 Charlottesville, VA 22903-2442
 (804) 982-2073 jfe7w@virginia.edu

Abstract

A global optimization algorithm which generalizes Kushner's (1964) univariate search is introduced. It aims to minimize the number of probes (function evaluations) required for a given level of certainty in the results. All known probes contribute to a stochastic model of the underlying "score surface"; this model is interrogated for the location with the highest probability of exceeding the current result goal. The surface is assumed to be characterized by Brownian motion, leading to a piecewise gaussian model, where the local regions are defined by the Delaunay triangulation of the probes. The algorithm balances the competing aims of 1) sampling in the vicinity of known peaks, and 2) exploring new regions. Preliminary tests on a standard 2-d search problem are very encouraging.

1. Introduction

Global search techniques are required to optimize parameters which nonlinearly affect the output of a model, such as with logistic regression, or the intermediate weights of an artificial neural network (ANN). Global search is also called for when the fitting criterion, or *score function*, is anything other than a few special accuracy metrics, such as mean squared error (MSE, L_2) or least absolute deviations (LAD, L_1). For instance, the cost of errors can be asymmetric (e.g., a classification "false alarm" can be much less costly than a "false dismissal") or, a range of estimated values might correctly lead to the same action (e.g., buy/sell), or the true score function might include objectives (low cost, high safety, etc.) other than accuracy alone. In fact, the main reason linear (L_2) models are employed so extensively in so many fields is their strong mathematical tractability and the mass of tools, training, and experience available as a result. The significant benefits of linear models should only reluctantly be abandoned, but *once set aside, the resulting freedom to design a score function to match the true use of the model should be exploited*. This requires global search.

2. Model-Based Search

Global optimization of parameters over a database of example cases can be likened to depth-sounding, as depicted in Figure 1. One probes for the depth (evaluates the score function) at a given location (set of parameter values) in pursuit of the deepest spot (global minimum). Model-based searches build up a rough picture of the ocean floor as probe

results become available, in order to home in on the goal as quickly as possible, and to have some confidence that a (reasonably) deeper point is not "out there somewhere".

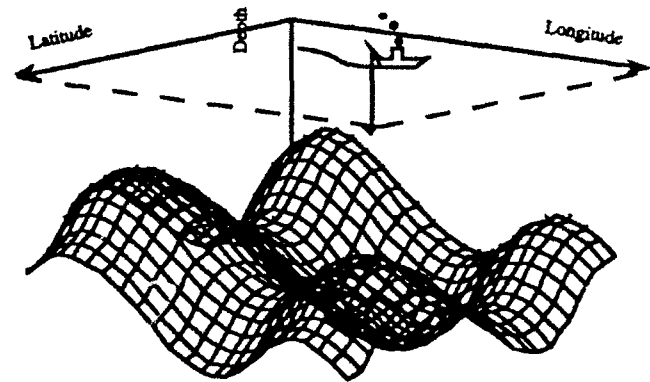


Figure 1 (after Elder and Finn, 1991):
 Global Optimization is Like Depth-Sounding;
 Choose a location (parameter setting) and probe for the depth
 (evaluate the function) looking for the minimum.

The surface model is interrogated for a good location to probe. Should analytic solutions prove too difficult, *internal model search* can be employed; that is, the model (as a rapid surrogate for the true score function) can be sampled at various candidate probe locations to determine the most promising one. New probe results then update the model, and the cycle continues.

The algorithm introduced here attempts to generalize the elegant 1-dimensional stochastic method of Kushner (1964) and was inspired by the continuing work toward this end of Stuckman and colleagues (e.g., Stuckman, 1988). The refinements of (Pertunen, 1991) and this paper are believed to lead to a d -dimensional algorithm more theoretically consistent (i.e., less heuristic) and hence, more efficient. The new algorithm is designed to require drastically fewer function evaluations than conventional searches. (It can be said to "think" more but "run" less than other methods.) It is capable of discovering multiple extrema whether or not the function is differentiable and, as it builds on known results, can be paused and restarted with no waste of probing effort. Importantly, a confidence measure in the results is provided, which is an approximation to the probability that the final answer could be improved upon by continued probing. Lastly, the algorithm is easily parallelized, with anticipated speed-up nearly linear in the number of added

processors. This paper describes Kushner's 1-dimensional method, the important Stuckman et al. generalizations, and the enhancements leading to the new algorithm, known as Global R^d Optimization when Probes are Expensive (GROPE).¹

3. Kushner's 1-Dimensional Search

Kushner's optimization method for one dimension models the score surface, y , as a *random walk* in x ; that is, as a *Brownian motion* or *Weiner process*, where the y value at a point x is distributed as a gaussian random variable with respect to neighboring points. An example "drunkard's walk" is shown in Figure 2; the path is like that of one moving forward (in x) at a constant rate, but staggering random amounts to each side. The stagger distribution is gaussian, and for a time series with discrete steps, y_{t+1} can be shown to be $N(y_t, \sigma^2)$. That is, the mean value for the next point is the current point; no information about the direction of the step is available. In particular, knowledge of earlier y values (i.e., how the curve got to y_t) is of no use; the distribution is *memoryless*, or *Markovian*, and the only values affecting the estimator of an unknown point are those of its nearest left and/or right neighbors in x .

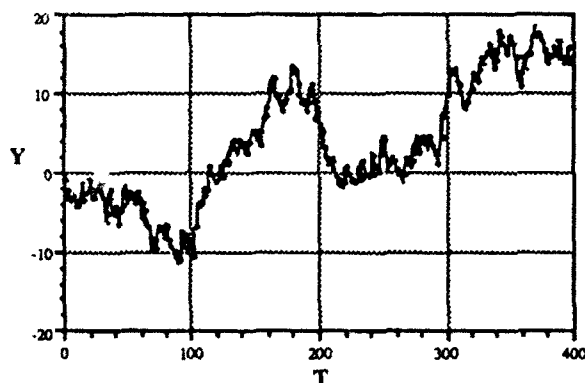


Figure 2: An Example Random Walk
($Y = \text{Cumulative Sum of } N(0,1) X_i$)

The random walk is descriptive of many processes, ranging from the Brownian motion of particles suspended in a liquid to the price history of the "Standard and Poor 500" index in the stock market. As a surface model, it has the advantage of being *fractal* or *locally rough* (it is nowhere differentiable) but *regionally smooth*. Thus, it is possible, though unlikely, for a large jump in y to occur between

points close in x ; but, most near differences will be small, and the surface is broadly "rolling" -- a representation capable of fitting many practically occurring functions.

A further practical advantage of the representation is its mathematical tractability. In the case of no noise, the Markovian property implies that the conditional expected value of y at a position x between two known points a and b , is the linear interpolant (shown in Figure 3a):

$$\mu(x | x, y) = y_a + p(y_b - y_a) \quad (1)$$

where the proportion p is $\frac{x-x_a}{x_b-x_a}$. Also, the variance conditioned on all previous results is a quadratic function of the distance from the interpolating bounds:

$$\sigma^2(x | x, y) = cp(1-p)(x_b - x_a) \quad (2)$$

for some slope factor, c (the mean squared variation in y as x changes). (Note that σ^2 has no other dependence on the y values.) As depicted in Figure 3b, this variance grows linearly with distance when only one neighbor is known (while the mean remains constant at the edges). When noise is present (i.e., probes at the same location can return different values), the representation is only slightly adjusted (Kushner, 1962): $\mu(x)$ does not go through the samples exactly, but *shrinks* toward neighboring samples, and $\sigma^2(x)$ is positive, not zero, at the probes.

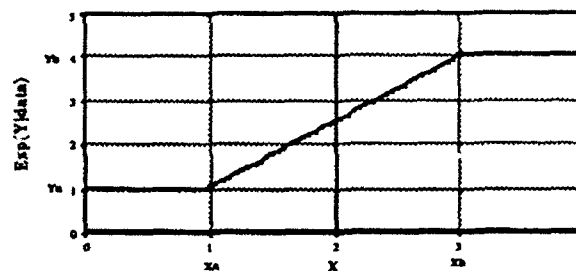


Figure 3a: Expectation of Y Conditioned on Y_a and Y_b

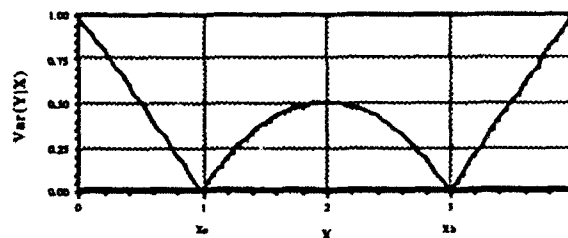


Figure 3b: Variance of Y Conditioned on Known Values at X_a and X_b

¹Earlier search algorithms by the author with that name are Guided Random Optimizer of Performance Error (e.g., Elder and Barron, 1988), and Global Regression of Probe Evaluations (Elder, 1991). Some acronyms never die!

Kushner (1964) solved for the probe location most likely to exceed the current best value by a fixed amount

(and suggested this magnitude could change with time, predating the effectively similar strategy of "temperature scheduling" which directs *simulated annealing*.) A slightly different perspective, proposed by Stuckman and Scannell (1991), is to seek the point most likely to exceed a given result goal, y_g ; that is, to find the x which maximizes

$$\Pr[y_x > y_g | x, y] = 1 - \Phi\left[\frac{y_g - \mu(x|y)}{\sigma(x|y)}\right] \quad (3)$$

This is depicted in Figure 4 for a one-dimensional line segment. Locations, x , close to that of b have the advantage, in putting probability mass across y_g , of starting closer (closer mean); yet, locations in the middle of the segment stretch farther (have greater standard deviations). Thus, the conflicting aims of exploration and exploitation are balanced. The goal-exceeding objective is also appealing when a natural bound is available, whether from known or theoretical limits (e.g., zero error) or, say, a competitor's results! When a value is not available however, the algorithm can employ the usual "carrot on a stick" approach, and strive to beat the current best result by a (possibly dwindling) amount.

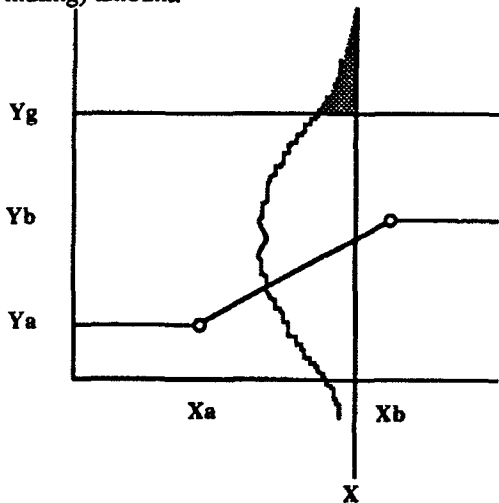


Figure 4: Maximize Probability of Exceeding Goal, Y_g
Given $Y(x) \sim N(\mu(x), \sigma(x))$

As the cumulative normal function, Φ , is monotonic, we may maximize (3) by *minimizing* its argument, or rather, the square of the argument, to accommodate the variance (2). Substituting, this translates to finding the proportion p which minimizes

$$A(p) = \frac{[y_g - (y_a + p(y_b - y_a))]^2}{cp(1-p)(x_b - x_a)} \quad (4)$$

Solving $\frac{\partial A}{\partial p} = 0$ reveals that the optimal location depends

only on the relative distance of the end points to the goal:

$$p^* = \frac{\Delta a}{\Delta a + \Delta b} \quad (5)$$

where $\Delta a = y_g - y_a$, $\Delta b = y_g - y_b$. (Note that the slope parameter, c , has no influence on p^* , and may be dropped.) The value $A(p^*)$ is a distance monotonic with the line segment's maximum conditional probability of containing a probe capable of exceeding the goal.

The 1-dimensional algorithm can be summarized:

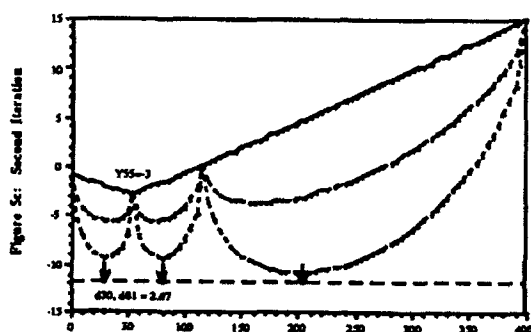
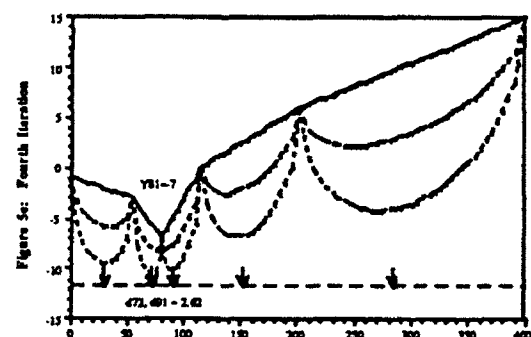
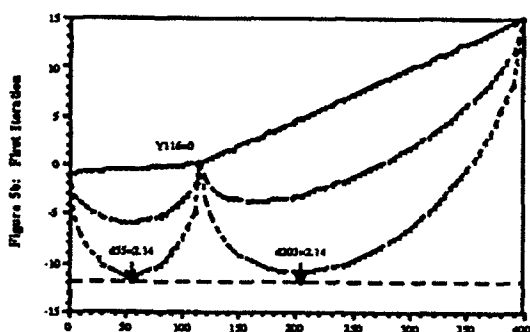
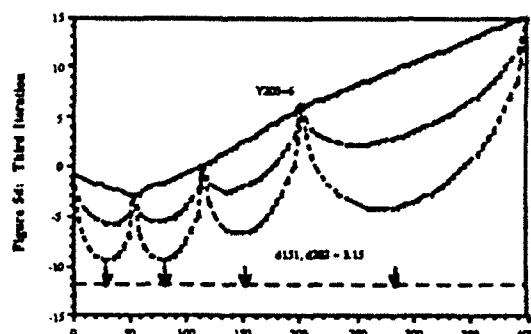
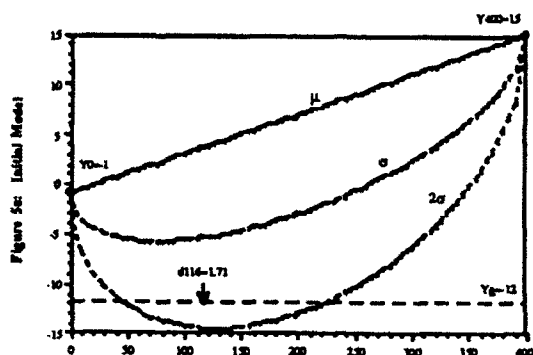
- 1) Probe the two bounds of the search space.²
- 2) Calculate the best sampling location, p^* , for the line segment, and insert that location, x^* , in an ordered list according to its distance estimate, $A(p^*)$.
- 3) Remove the top member of the list and probe at x , breaking the line segment (a, b) into (a, x) and (x, b) .
- 4) Repeat steps 2 and 3 (with two segments for step 2) until the goal is exceeded, resources are exhausted, or the probability of better results is acceptably low.

A few steps of the algorithm are shown in Figure 5.

After each probe, the list of candidate probe locations increments in length, as one segment is removed and two take its place. Unless the goal changes, in which case the optimal location for each segment must be reevaluated, only the p^* locations for the two new segments must be computed. (And it can be shown that their two distances are identical; e.g., Figure 5b. In that example, ties were broken randomly.) When results are far from the goal (presumably at the beginning of the search), the variance component of (4) dominates and locations relatively midway between known probes are preferred. When the best probes instead score near the goal, further probes in their vicinities are called for. This mechanism performs the tradeoff between the conflicting search aims of 1) homing in on the extreme of a promising area, and 2) sampling unknown regions.

The search is terminated if either the goal or the probe limit is reached. Alternately, the slope parameter c (usually not known) can be estimated from the probe results, and used to compute the probability, according to the underlying model, that some probe yet exists which could exceed the goal. In a similar manner, Stuckman (1988) employs \hat{c} , the maximum likelihood estimator (MLE) of c , along with the single closest candidate probe, to calculate the probability that the *next* guess will do the job -- stopping when this

²The initializing probes can be at locations other than the bounds (in which case, the algorithm would begin with three line segments). However, if the first samples are not close to the edges of the legal space, the algorithm may later call for probes there anyway, since the variance grows so rapidly beyond the outermost probe.



value is very small, e.g. 10^{-6} . However, as each candidate probe is independent under the random walk model, one may calculate the *joint* probability that one of the N current candidates could exceed the goal,

$$\Pr[\exists i \ni y(p_i) > y_g \mid \mathbf{p}, A(\mathbf{p})] = 1 - \prod_{i=1}^N \Phi[\sqrt{A(p_i)}] \quad (6)$$

where \hat{c} is used in A (equation 4).

A more reliable analysis of joint probability would be *predictive* (Aitchison and Dunsmore, 1975), and employ the

full distribution of c . Such an estimator weights each possible c value by its relative effect on the likelihood of the data, $L(\mathbf{x}|c)$. In place of $A(\mathbf{x})$ with \hat{c} , one would use

$$\int_C A(\mathbf{x}, c) p(c|\mathbf{x}) dc, \text{ where } p(c|\mathbf{x}) = \frac{L(\mathbf{x}|c)}{\int_C L(\mathbf{x}|c) dc} \quad (7)$$

This is superior to the *estimative* technique of (6), which acts as if all the mass of the likelihood were on the MLE mode, $\hat{c} = \sup(L(\mathbf{x}|c))$. Computations are substantially facilitated by employing *conjugate prior* distributions for the parameters (as described in the above reference).

4. Extension to Multiple Dimensions

The key difficulty in expanding Kushner's algorithm from R^1 to R^d -- and perhaps the reason the method saw little use for a generation -- is the extension of the random walk model into a random *field*. There are two field definitions in the literature (e.g., Adler, 1981) and, an approximation to a third is employed here.

The *multi-univariate* version of Kushner's method (Stuckman, 1988) avoids the issue of random fields, and

instead employs the one-dimensional algorithm along the line segments connecting all pairs of probes (or a subset of k -nearest neighbors for each probe, where k jumps an order of magnitude when the probe is the current best). However, such a technique can ignore a probe intermediate to another pair and, more importantly, is silent about function values everywhere except the line segments connecting probes.

To cover the search space, the region within the *convex hull* of the probes can be *tessellated* (divided into space-filling disjoint regions) into a set of *simplices*. In R^d , a simplex is a convex polyhedron composed of $d+1$ vertices (i.e., a triangle in two dimensions; a tetrahedron, in three). If a *simplex subdivision* approach is employed (e.g., Groch et al., 1985), a new probe divides its surrounding simplex into $d+1$ smaller simplices (defined by the new point and each *face* of the old simplex), leaving all other regions intact. It would be better, however, to update the entire tessellation in a manner maintaining some optimality property, such as *local equi-angularity*, as proposed by Lawson (1972), in which small angles in triangles are avoided. Sibson (1978) proved that the unique set of connections with this property in the plane is the *Delaunay triangulation*. (However, in three and higher dimensions, this triangulation does not necessarily maximize the minimum *solid angle* (Joe, 1989).)

The Delaunay triangulation is the dual of the *Voronoi* (or Dirichlet or Thiessen) tessellation, wherein regions are partitioned according to the nearest neighbor rule. That is, all the points within a region are closer (by L_2) to the same known probe than they are to any other. Another property of the triangulation has long been known for low dimension (Miles, 1970), but only recently proven in general (Rajan, 1991): the *circumscribing sphere* of each simplex is empty; i.e., the only triangulation in which the sphere intersecting the vertices of a given simplex contains no other point, is the Delaunay (a construction illustrated in Figure 6). (Rajan further proved that the maximum radius of the smallest of such spheres is less than that for any other triangulation -- allowing the Delaunay triangulation to be formulated as a solution to a continuum optimization problem.)

The properties of Delaunay triangulations make them useful to many disciplines, including (in 2-d): surface interpolation, geophysical contouring, and the study of the spread of epidemics; and 3-d modelling of the interface network of polycrystalline materials. The optimization algorithm of Perttunen (1991) employs Delaunay triangulation to tessellate the search space, but scores each candidate simplex with a heuristic, nonparametric metric: the product of the *ranks* of the vertices divided by its content, or "hypervolume" (Perttunen and Stuckman, 1990). The next probe is taken within the winning simplex, at the weighted average location of its vertices (where the weights are the inverse relative ranks of the probe scores).

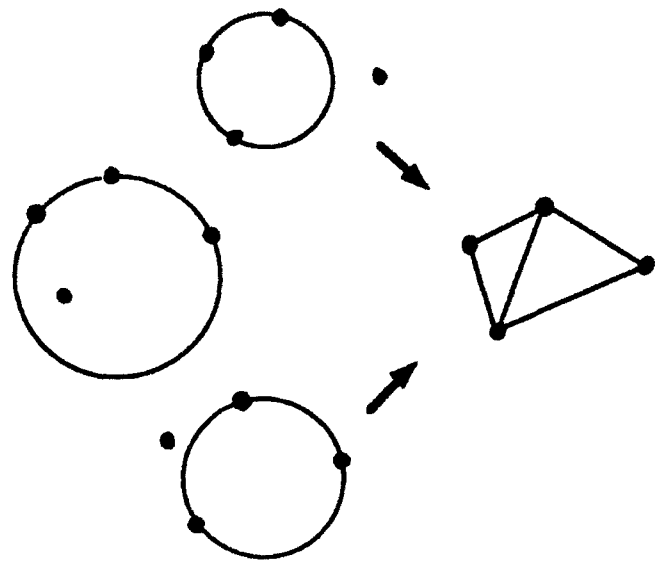


Figure 6: Delaunay Triangulation of 4 Points according to Empty Circumsphere Rule

Incorporation of Delaunay triangulation improved the ranking method, causing the scatter plot of search points to better correspond to the contour diagram of each objective function tested (Perttunen, 1991). However, use of the ranking, coverage, and weighting heuristics lead to a technique having little in common with Kushner's stochastic algorithm. Building on the ideas of tessellation and goal-direction however, a more straightforward generalization is possible. One could perform a linear interpolation of the response values at the Delaunay vertices to define the conditional expected values inside a simplex, and use a quadratic polynomial for the conditional variance, constrained to agree with Kushner's quadratic variance curve along each 1-dimensional simplex edge. The expectation is thus composed of a piecewise planar surface -- resembling facets of a gem (or, perhaps, the *hinging hyperplane* modelling technique recently proposed by Breiman, 1991). For example, the 2-dimensional Delaunay triangulation of probes in Figure 7, leads to the interpolation surface illustrated in Figure 8.

The relative variance "canopy" arches over the simplex as shown in Figure 9, from lows at locations with known values, to an interior peak far from the vertices. This variance can be defined by the (unique) complete quadratic polynomial in d variables which conforms to Kushner's equations along the $\binom{d+1}{2}$ edges of the simplex (and is undefined outside these bounds). These variance constraints are imposed since a hyperplane defined by the vertex values is used for the expectation. The variance can be viewed as a measure of uncertainty about the mean, so their methods of

estimation must be linked. Along an edge, only the pair of connected vertices affect the conditional mean value³ (as with Kushner's 1-dimensional method); therefore, the edge constraints on variance are *necessary* for this generalization of the algorithm.

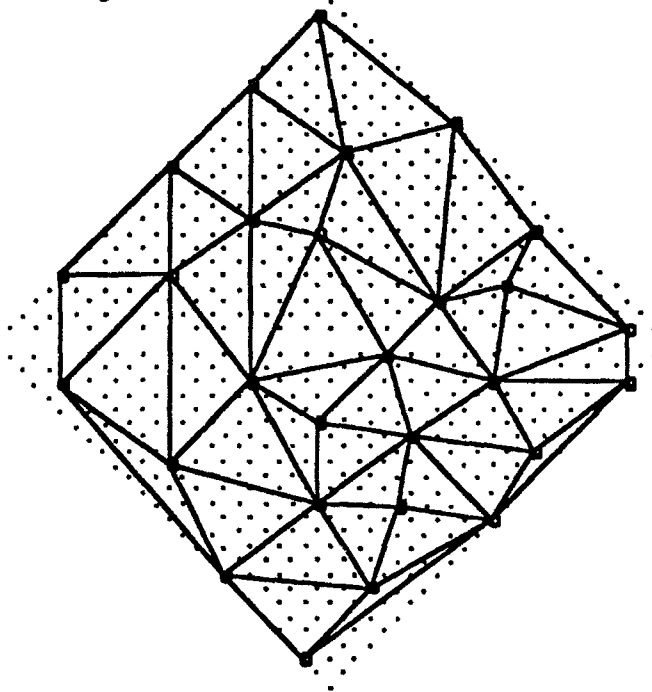


Figure 7: Delaunay Triangulation of 28 Probes
(from grid of 25x26 potential sites)

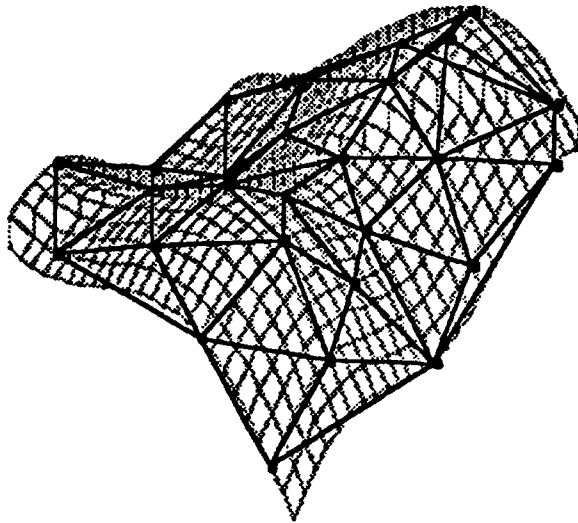


Figure 8: Triangular Facets Interpolate Function Surface

³This 1-dimensional property can have the side-effect of ignoring the single nearest known probe, as can happen (for a "thin" simplex) when estimating the values of an edge segment near a third vertex. This behavior however, is confined primarily to the convex hull (i.e., outer edges) of the space.

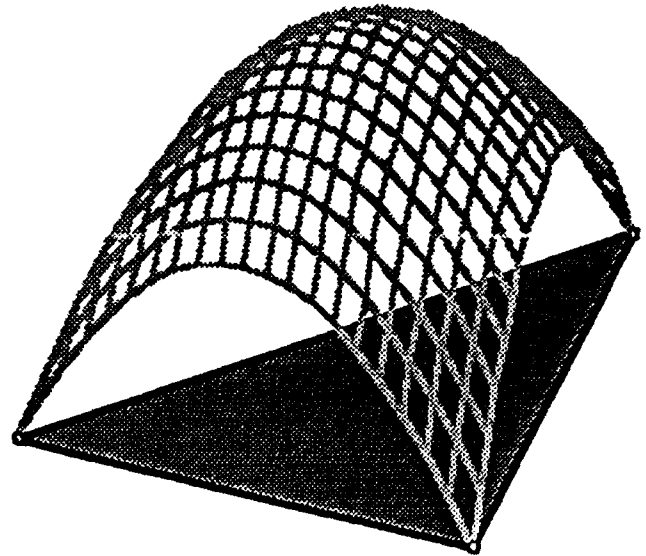


Figure 9: Example Variance Canopy
(Constrained to agree with Kushner's quadratic equation
along 1-dimensional edges of the simplex)

The edge constraints are also *sufficient*. There are $\binom{d+2}{2}$ parameters in a second-order polynomial in d variables. (In general, the complete polynomial with maximum power p has $\binom{d+p}{p}$ terms). Each of the $\binom{d+1}{2}$ edges of the simplex contributes one constraint -- say, the value of the variance at the midpoint of the segment (which, for a given c , is defined by its length (2)). The remaining $d+1$ constraints are provided by the vertices of the simplex, for which the variance is a minimum fixed value (zero for noiseless probing). Since none of the edges are collinear (as demonstrated by the existence of a circumsphere for the simplex), the exact match of constraints and degrees of freedom means the polynomial solution will be unique and have zero error. (Still, in practice, thin triangles on the convex hull can lead to nearly collinear edges. Thus, robust regression techniques (e.g., singular value decomposition) which remove near-singularities are required.

The locations and scores of the $d+1$ probes of each simplex thus define the equations for the linear expectation, $\mu(x)$, and quadratic variance, $\sigma^2(x)$, of its interior (which may be solved for using ordinary regression). In one dimension, the optimal interior location, x^* , for each simplex is known analytically (5). This can also be shown to be the case for two dimensions, but the solution is surprisingly complex. For multidimensional applications, an easier approach is to perform an internal search of the function to minimize

$$A(x) = \frac{(y_g - \mu(x))^2}{\sigma^2(x)} \quad (8)$$

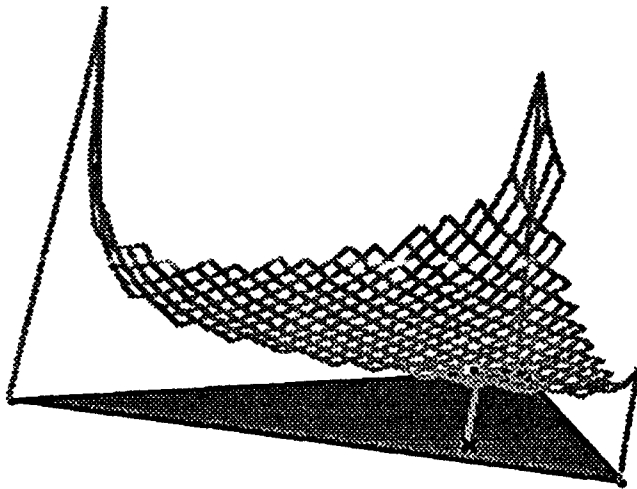


Figure 10: Example $A(x)$ Surface
(with Location of Minimum Noted)

As shown for a 2-dimensional example in Figure 10, this squared distance function is positive, smooth, and unimodal -- allowing any of several local minimizers to be employed. (However, the function is not defined outside the simplex, and explodes at the vertices, so care must be taken at the boundaries.)

The GROPE algorithm is initialized by probing $d+1$ points from the convex hull defining the search space, or by absorbing previous results. (As all probes contribute to the model locally, any restart of the program can pick up exactly where a prior run left off.) Then, until the goal, y_g , is reached, resources run out, or the probability of improvement is sufficiently slight (see above), iteratively:

1. Construct/Update the Delaunay triangulation, removing candidate probe locations representing obsolete simplices from the ordered list.
2. For each new simplex j :
 - a) Solve for $\mu_j(x)$ given vertices.
 - b) Solve for $\sigma_j^2(x)$ given vertices and edges.
 - c) Find the best probe location, x_j^* , for the simplex by minimizing $A_j(x)$ (8) (the squared, standardized distance to the goal).
 - d) Insert this candidate probe location into a list ordered by $A_j(x)$.
3. If locations on the intended convex hull remain unknown, probe there; otherwise, pop the head of the list, and probe at that location.

Steps 1 and 2c, the re-triangulation and the internal search of new simplices, are most affected by the number of probes, N , and problem dimension d . The added overhead is rather great (compared even to some other model-based searches). However, whenever probe computations are not

trivial, that time should be more than compensated for by the algorithm's judicious choice of locations. (Time saved not "running" > Extra time spent "thinking"). For example, in this author's aerospace experience (e.g., Elder and Barron, 1988), each probe for a guidance or control application consisted in running a full computer simulation with a new set of parameters. Such a task can easily take minutes per probe on a workstation (and engender, in the early morning hours, a visceral distaste for senseless search methods!).

If q processors are available (and if the application permits) $q - 1$ probes may be removed from the head of the list and evaluated in step 3. The last processor could update the Delaunay triangulation, given the locations of the new probes, as the triangulation does not depend on their results, y . Of course, the problem addressed should be reformulated to: find the best *set* of probes such that one is likely to exceed the goal. However, this simple q -at-a-time method should provide near-linear speedup for a common type of hard problem, where many regions of the domain must be explored (in which case the probes may as well be simultaneous as sequential).

5. Early Experimental Results

A 2-dimensional prototype of GROPE has been prepared, which uses Tipper's (1991) program for planar Voronoi tessellation (mapped into Delaunay triangles), and the *downhill simplex* method of function minimization (Nelder and Mead, 1965) for the internal model search as programmed by (Press et al., 1988). The test function was the bimodal "Hosaki" equation (Bekey and Ung, 1974)

$$(1 - 8x_1 + 7x_1^2 - \frac{7x_1^3}{3} + \frac{x_1^4}{4}) x_2^2 \exp(-x_2) \quad (9)$$

pictured in Figure 11. The global minimum for $x_1 \in [0,5]$, $x_2 \in [0,6]$ is -2.345 at $x = (4,2)$.

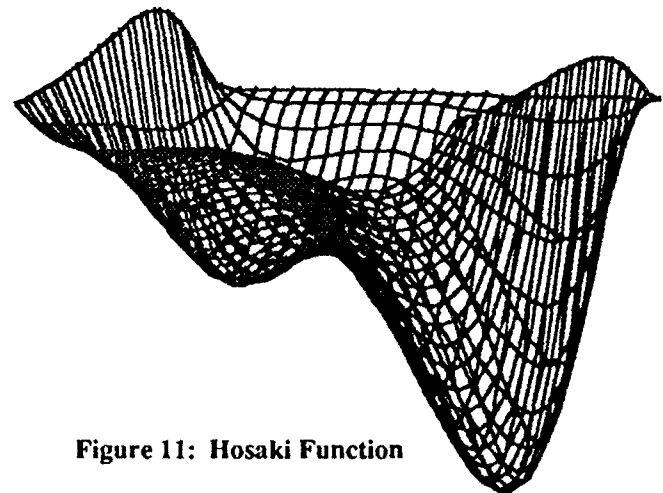


Figure 11: Hosaki Function

For testing purposes, nearness to the final answer was the stopping criterion. This is not usually possible for "field" applications, but allowed comparison with two random methods: *modified random creep* (Bekey and Ung, 1974) and *adaptive random search* (Ponzato et al., 1984). In addition, recent results from the *sequential design for optimization*, a promising RBF-like model-based technique of Cox and John (1992) are included in Table 1.

Table 1: Hosaki 2-d Function Results

Method	#Probes to Soln.
Modified Random Creep	451
Adaptive Random Search	830
Sequential Design for Optimization	55 (constant param.) 36 (linear param.)
GROPE (2nd trial)	11 (goal = -3.0)

The first GROPE run employed the known minimum as the goal, y_g , but crept too cautiously toward the final location, and was abandoned.⁴ The cautious approach suggests that local probing is overly preferred to exploration of new areas; i.e., that the role of variance is too low relative to that of expectation. Accordingly, a more remote goal, $y_g = -3$, was set (though, of course, still halting at y_{min}), leading to much improved results: only 11 probes. The final triangulation of this second run is pictured in Figure 12, where the vertices of the Delaunay triangulation are known probes, each "x" represents a candidate probe location for its triangle, and the "." denote discarded candidate locations (due to dissolution of the surrounding triangle). The position of the global minimum is noted ("."), and the 11th probe value was -2.344. Note that the number of triangles (and thus candidate locations) increases by two after each probe -- a property of the 2-d Delaunay triangulation.

6. Potential Improvements

Both GROPE runs in the example problem were initialized by probing the four corners of the search domain. Such rectangular bounding requires 2^d initial probes, which can be expensive in problems of high dimension. Furthermore, these first probes are taken in the regions least expected to produce useful results: the domain boundaries. A minimum of $d+1$ probes (a single initial simplex) can define the domain; yet, to roughly match the content of the hyper-

rectangle, these probes would have to be even more extreme in location. It is convenient for the algorithm to have the bounds set initially, and always be performing a type of interpolation operation. But instead, perhaps some type of Bayesian technique, with distributions reflecting the desirability of probing in the center of the region, could be employed. This could restrain, to an adjustable degree, the otherwise linearly increasing variance beyond the outermost probe towards the bounds.

A simpler improvement, as suggested by the example application, would be to adjust the single current parameter of the algorithm (y_g) with time and/or performance -- a "relaxation" technique similar to other methods. Further research may show that a good goal scheduling strategy can be inferred for a problem from metrics of its ongoing results -- e.g., the distribution of probe results and its extreme, the (estimated) smoothness of the score surface and its variability, and the distribution of simplex content.

Also, as experienced in the first trial, regression singularities can occur when fitting the variance of long "thin" triangles near the convex hull having nearly collinear edges. Use of a robust fitting method (e.g. singular value decomposition with removal of small eigenvalues) is being investigated to remove this symptom of "overfit".

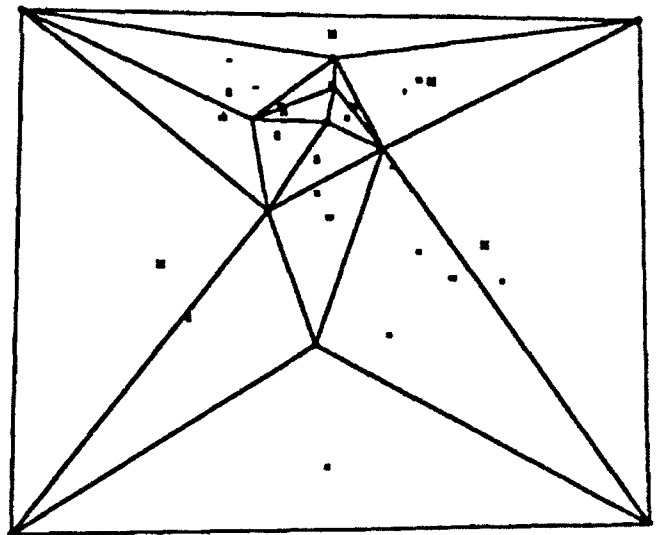


Figure 12: Triangulation of Hosaki Function after 11th Probe

7. Conclusions

The GROPE algorithm is a novel, efficient, model-based stochastic R^d optimizer, in large part generalizing Kushner's elegant 1-dimensional method. For medium-dimensioned problems (up to a dozen variables, say), the model-based search technique should provide more accurate results using (drastically) fewer probes than competing methods, and provide an interpretable confidence in the outcome.

⁴The fractal surface assumed by the algorithm is unlike the smoothness exhibited by the "toy" test problem; however, the main problem encountered was nearly collinear vertex edges, which will require a robust (internal) regression method.

REFERENCES

- Adler, R.J. (1981). *The Geometry of Random Fields*. Wiley, New York.
- Aitchison, J., I.R. Dunsmore (1975). *Statistical Prediction Analysis*. Cambridge Press, London.
- Bekey, G.A., M.T. Ung (1974). A Comparative Evaluation of Two Global Search Algorithms, *IEEE Trans. SMC* 4: 112-116.
- Breiman, L. (1991). Hinging Hyperplanes for Regression, Classification, and Function Approximation, *Tech. Rpt. 324*, Dept. Statistics, UC Berkeley, CA.
- Cox, D.D., S. John (1992). A Statistical Method for Global Optimization, *Tech. Report*, Dept. Statistics, Univ. Illinois (Champaign), April.
- Elder, J.F. IV, R.L. Barron (1988). Automated Design of Continuously-Adaptive Control: The "Super-Controller" Strategy for Reconfigurable Systems, *Proc. American Control Conf.*, Atlanta, GA, June 15-17.
- Elder, J.F. IV (1991). Global Regression of Probe Evaluations, *3rd Int'l Workshop on Artificial Intelligence and Statistics* (extended abstract), Ft. Lauderdale, Florida, Jan. 2-5.
- Elder, J.F. IV, M.T. Finn (1991). Creating 'Optimally Complex' Models for Forecasting. *Financial Analysts Journal*, Jan/Feb, pp. 73-79.
- Groch, A., L.M. Vidigal, S.W. Director (1985). A New Global Optimization Method for Electronic Circuit Design, *IEEE Trans. Circuits and Systems* 32, no. 2.
- Joe, B. (1989). Three-Dimensional Triangulations from Local Transformations, *SIAM J. Sci. Stat. Com.* 7: 514-539.
- Kushner, H.J. (1962). Stochastic Model of an Unknown Function, *Journal of Mathematical Analysis and Applications* 5: 150-167.
- Kushner, H.J. (1964). A New Method of Locating the Maximum of an Arbitrary Multipeak Curve in the Presence of Noise, *Journal of Basic Engineering*, March, pp. 97-106.
- Lawson, C.L. (1972). Generation of a Triangular Grid with Application to Contour Plotting, *CIT Jet Propulsion Laboratory, Tech. Mem.* 299.
- Miles, R.E. (1970). On the Homogeneous Planar Poisson Point Process, *Mathematical Biosciences* 6: 85-127.
- Nelder, J.A., R. Mead (1965). A Simplex Method for Function Minimization, *Computer Journal* 7: 308-313.
- Perttunen, C.D., B.E. Stuckman (1990). The Rank Transformation Applied to a Multiunivariate Method of Global Optimization, *IEEE Trans. SMC* 20, no. 5: 1216-1220.
- Perttunen, C.D. (1991). A Computational Geometric Approach to Feasible Region Division in Constrained Global Optimization, *Proc. IEEE SMC Conf.*, Charlottesville, Virginia, Oct. 13-16.
- Press, W.H., B.P. Flannery, S.A. Teukolsky, W.T. Vetterling (1988). *Numerical Recipes in C*. Cambridge Univ. Press, New York.
- Pronzato, L., E. Walter, A. Venot, J.F. Lebruchec (1984). A General Purpose Global Optimizer: Implementation and Applications, *Mathematics and Computers in Simulation* 26: 412-422.
- Rajan, V.T. (1991). Optimality of the Delaunay Triangulation in R^d , *Proc. 7th ACM Symposium on Computational Geometry*: 357-363.
- Sibson, R. (1978). Locally Equiangular Triangulations, *Computer Journal* 21, no. 3: 243-245.
- Stuckman, B.E. (1988). A Global Search Method for Optimizing Nonlinear Systems, *IEEE Trans. SMC* 18, no. 6: 965-977.
- Stuckman, B.E., P. Scannell (1991). A Multidimensional Bayesian Global Search Method which Incorporates Knowledge of an Upper Bound, *Proc. IEEE SMC Conf.*, Charlottesville, Virginia, Oct. 13-16.
- Tipper, J.C. (1991). FORTRAN Programs to Construct the Planar Voronoi Diagram, *Computers and Geosciences* 17 no. 5: 597-632.

Multivariate Stable Distributions

John P. Nolan

Department of Mathematics and Statistics

American University

Washington, DC 20016

e-mail: jpnolan@american.edu

Abstract

Stable distributions can be used to model phenomena where the underlying distribution has heavy tails. A difficulty in using stable distributions in applications is that, except for a few special cases, there is no explicit formula for the densities. This paper describes how an arbitrary stable distribution can be approximated by a stable distribution with a discrete spectral measure. Densities of the approximation may be numerically computed by inversion of the characteristic function and a method is discussed for generating stable random vectors for simulation purposes.

1. Introduction

Let $\vec{X} = (X_1, \dots, X_d)$ be a d -dimensional α -stable random vector, $0 < \alpha \leq 2$. The best known examples are normal ($\alpha = 2$) and Cauchy ($\alpha = 1$) random vectors; for general definitions and theory see Samorodnitsky and Taqqu (1992). The distribution of \vec{X} can be described in terms of a spectral measure in the following way. Let $\phi(\vec{t}) = \mathbb{E} \exp(i\langle \vec{t}, \vec{X} \rangle) = \mathbb{E} \exp(i \sum_{j=1}^d t_j X_j)$ be the joint characteristic function of \vec{X} . Kuelbs (1973) showed that \vec{X} is α -stable if and only if there exists a finite measure σ on the unit sphere $S^{d-1} \subset \mathbb{R}^d$ and a shift vector $\vec{\mu} \in \mathbb{R}^d$ such that $\phi(\vec{t}) = \exp(-\int_{S^{d-1}} \psi(\langle \vec{t}, \vec{s} \rangle) \sigma(d\vec{s}) + i\langle \vec{t}, \vec{\mu} \rangle)$, where

$$\psi(t) = \begin{cases} |t|^\alpha (1 - i \tan \frac{\pi\alpha}{2} \text{sign}(t)) & \alpha \neq 1 \\ |t|(1 + i \frac{2}{\pi} \text{sign}(t) \ln |t|) & \alpha = 1, \end{cases}$$

Throughout we assume that \vec{X} is truly d -dimensional, which is equivalent to the support of σ spanning \mathbb{R}^d . We will also assume that the shift vector $\vec{\mu}$ is zero.

2. Approximation

This section will show how to approximate the density $p(\vec{x})$ of \vec{X} in terms of the spectral measure σ . Theorem 1 shows we can approximate $p(\vec{x})$ by a computationally

simpler density, one that corresponds to a stable distribution with a discrete spectral measure. To state the theorem we need a definition. Given a finite partition A_1, \dots, A_n of S^{d-1} and points $\vec{s}_1, \dots, \vec{s}_n$ with $\vec{s}_j \in A_j$, define a discrete measure σ^* based on σ by concentrating mass $\sigma(A_j)$ at \vec{s}_j , i.e.

$$\sigma^*(\cdot) = \sum_{j=1}^n \sigma(A_j) \delta_{\vec{s}_j}(\cdot) \quad (2.1)$$

Theorem 1 Let \vec{X} be a truly d -dimensional α -stable random vector ($d \geq 2, 0 < \alpha < 2$) with spectral measure σ and density $p(\vec{x})$.

(i) For all $\epsilon > 0$, there is a discrete measure σ^* of form (2.1) which corresponds to a d -dimensional α -stable random vector \vec{X}^* which has a density $p^*(\vec{x})$ satisfying

$$\sup_{\vec{x} \in \mathbb{R}^d} |p(\vec{x}) - p^*(\vec{x})| \leq \epsilon.$$

(ii) For all $\epsilon > 0$, there is a discrete measure σ^* of form (2.1) which corresponds to a d -dimensional α -stable random vector \vec{X}^* which satisfies

$$\sup_{A \in \text{Borel}(\mathbb{R}^d)} |P(\vec{X} \in A) - P(\vec{X}^* \in A)| \leq \epsilon.$$

The proof of Theorem 1 can be found in Byczkowski, Nolan and Rajput (1991). It shows that the only requirement of the partition used in (2.1) is that the diameter of the sets is sufficiently small. An explicit value for this diameter (as a function of $\epsilon, \alpha, d, \sigma$), hence a concrete value for the number of terms in (2.1), is given in the reference. The method of proof is straightforward: if σ^* is "close" to σ , then the integrals in the exponents of $\phi^*(\vec{t})$ and $\phi(\vec{t})$ will be close. By the inversion formula for characteristic functions, the densities will be uniformly close.

3. Numerical Calculation

Theorem 1 above shows that an arbitrary stable density can be approximated by one with a discrete spectral

density. Thus we can understand the general behavior of stable densities by examining ones with discrete spectral measures. The characteristic function of a stable random vector having discrete spectral measure (2.1) is $\phi(\vec{t}) = \exp(-I(\vec{t}))$, where

$$I(\vec{t}) = \sum_{j=1}^n \psi(\langle \vec{t}, \vec{s}_j \rangle) \sigma(A_j).$$

Note that since σ is discrete, $I(\vec{t})$ is a finite sum. Hence the characteristic function is much simpler to evaluate than it would be if σ was arbitrary and the exponent of $\phi(\vec{t})$ required an evaluation of an integral over S^{d-1} .

The steps necessary to numerically invert this characteristic function are outlined below. Use the fact that $p(\cdot)$ is real to write

$$p(\vec{x}) = (2\pi)^{-d} \int_{\mathbb{R}^d} e^{-i\langle \vec{x}, \vec{t} \rangle} \phi(\vec{t}) d\vec{t} = \int_{\mathbb{R}^d} J(\vec{t}, \vec{x}) d\vec{t},$$

where $J(\vec{t}, \vec{x}) = (2\pi)^{-d} \exp(-\Re I(\vec{t})) \cos(\langle \vec{x}, \vec{t} \rangle + \Im I(\vec{t}))$. To approximate $p(\vec{x})$ to within ϵ , first truncate the region of integration to a disk: Lemma 7 of Nolan and Rajput (1992) gives an explicit bound for $K = K(\epsilon, \alpha, \sigma, d)$ such that

$$\left| \int_{\mathbb{R}^d} J(\vec{t}, \vec{x}) d\vec{t} - \int_{|\vec{t}| \leq K} J(\vec{t}, \vec{x}) d\vec{t} \right| \leq \epsilon/2.$$

What remains is a numerical integration problem: evaluate the d -dimensional integral of $J(\vec{t}, \vec{x})$ over the disk $|\vec{t}| \leq K$ to within $\epsilon/2$.

Nolan and Rajput (1992) give a program to calculate this integral when $d = 2$ that uses a 2-dimensional adaptive integration technique. Figure 1 shows both the density surface and the level contours of a density computed using this algorithm on a 41×41 grid. Clearly stable densities when $\alpha < 2$ can be very different from the elliptically contoured Gaussian densities.

4. Simulation

A method is described for generating stable random vectors \vec{X} having discrete spectral measure. One use for these random vectors is to test the robustness of multivariate statistical procedures: generate a data set with random noise having heavy tails and dependent components and evaluate how well a procedure performs. Another possible use is in calculating $P(\vec{X} \in A)$ for sets $A \subset \mathbb{R}^d$. Since the numerical calculation of these probabilities is difficult when $d > 2$, one can estimate them by standard Monte Carlo methods once we know how to generate vectors with the prescribed distribution. A

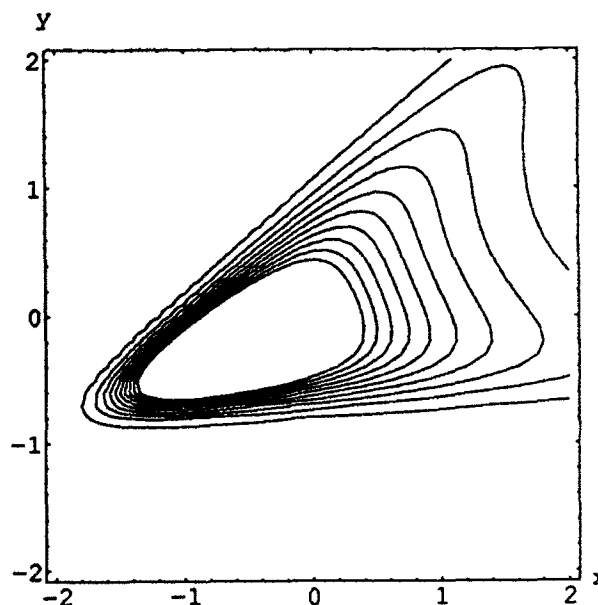
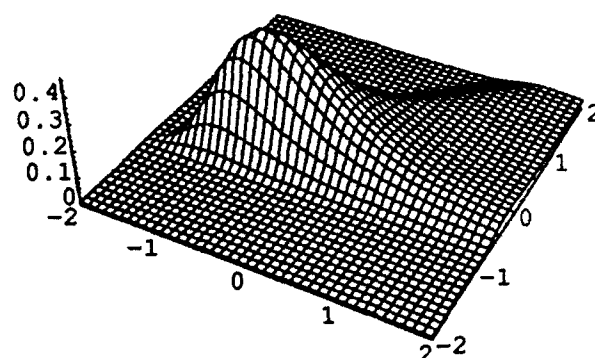


Figure 1: Density surface and level contours of a stable distribution. $\alpha = 1.25$ and the spectral measure has 3 point masses: $\sigma_1 = \sigma_2 = \sigma_3 = 0.2$ at $\vec{s}_1 = (1, 0)$, $\vec{s}_2 = (\cos \frac{2\pi}{3}, \sin \frac{2\pi}{3})$, and $\vec{s}_3 = (\cos \frac{4\pi}{3}, \sin \frac{4\pi}{3})$.

third use for such random vectors is in economic simulation, e.g. the stable portfolio analysis of Press (1972).

Let V be a one dimensional α -stable random variable that is normalized, has zero shift parameter and totally skewed to the right, i.e. the characteristic function of V is $\mathbb{E} \exp(itV) = \exp(-\psi(t))$. The following lemma says that any α -stable random vector with discrete spectral measure has the same distribution as a linear combination of vector multiples of such one dimensional i.i.d. stable random variables.

Lemma 1 Let $0 < \alpha < 2$, $\sigma_1, \dots, \sigma_n > 0$, $\bar{s}_1, \dots, \bar{s}_n \in S^{d-1}$, and V_1, \dots, V_n be i.i.d. one dimensional normalized α -stable random variables that are totally skewed to the right. If \bar{X} is the α -stable random vector with discrete spectral measure $\sigma(d\bar{s}) = \sum_{j=1}^n \sigma_j \delta_{\bar{s}_j}(\bar{s})$ and zero shift vector, then

$$\bar{X} \stackrel{d}{=} \begin{cases} \sum_{j=1}^n \sigma_j^{1/\alpha} V_j \bar{s}_j & \alpha \neq 1 \\ \sum_{j=1}^n \sigma_j (V_j + \frac{2}{\pi} \ln \sigma_j) \bar{s}_j & \alpha = 1. \end{cases} \quad (4.1)$$

Proof Note that for $r > 0$,

$$\psi(rt) = \begin{cases} r^\alpha \psi(t) & \alpha \neq 1 \\ r\psi(t) + i\frac{2}{\pi}(r \ln r)t & \alpha = 1. \end{cases} \quad (4.2)$$

First consider the case when $\alpha \neq 1$. Using (4.2), independence and the characteristic function of the V_j 's, the characteristic function of \bar{X} is

$$\begin{aligned} \mathbb{E} \exp(i \sum_{j=1}^n \langle \bar{t}, \sigma_j^{1/\alpha} V_j \bar{s}_j \rangle) &= \prod_{j=1}^n \mathbb{E} \exp(i \langle \bar{t}, \sigma_j^{1/\alpha} V_j \bar{s}_j \rangle) \\ &= \prod_{j=1}^n \exp(-\psi(\langle \bar{t}, \sigma_j^{1/\alpha} \bar{s}_j \rangle)) \\ &= \prod_{j=1}^n \exp(-\psi(\langle \bar{t}, \bar{s}_j \rangle) \sigma_j) \\ &= \exp(-\sum_{j=1}^n \psi(\langle \bar{t}, \bar{s}_j \rangle) \sigma_j). \end{aligned}$$

This is the result when $\alpha \neq 1$. The $\alpha = 1$ case is similar, using independence and (4.2).

The method of generating stable random vectors is straightforward: generate the one dimensional stable

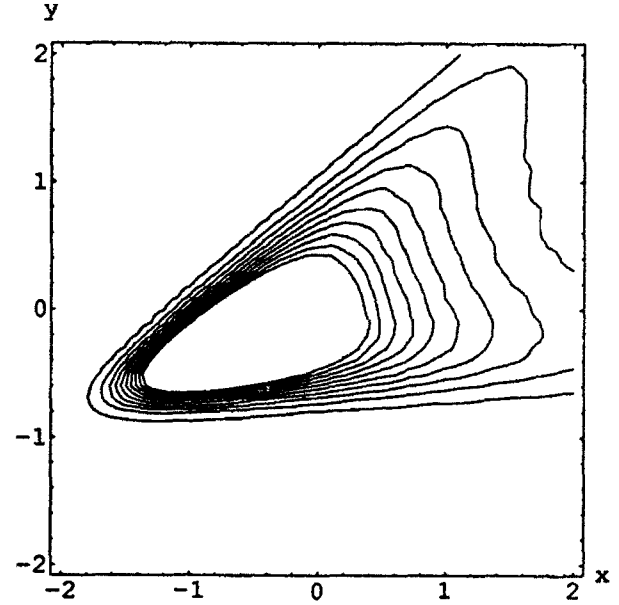


Figure 2: Contours of empirical density for randomly generated stable vectors with the same parameters as Figure 1.

random variables V_1, \dots, V_n and substitute into (4.1). Chambers, Mallows and Stuck (1976) gave an algorithm for generating such one dimensional stable variates. More details, a discussion of a geometric interpretation of Lemma 1, and the listing of a program that generates such random vectors is given in Modarres and Nolan (1992). Figure 1 is a plot of the level contours of the empirical density function generated by the method above. It is based on a simulation with 5,000,000 vectors.

5. References

- Byczkowski, T., Nolan, J. P. and Rajput, B. (1991), "Approximation of Multidimensional Stable Densities." *UNC Center for Stochastic Processes Technical Report 351*.
- Chambers, J. M., Mallows, C. L. and Stuck, B. W. (1976), "A method for simulating stable random variables." *JASA* 71,340-344. Correction *JASA* 82, 704 (1987).
- Kuelbs, J. (1973), "A representation theorem for symmetric stable processes and stable measures on H ." *Z. Wahr. verw. Geb.* 26, 259-271.
- Modarres, R. and Nolan, J. P. (1992), "A method for simulating stable random vectors." Preprint.

Nolan, J. P. and Rajput, B. S. (1992), "Numerical calculation of multidimensional stable densities." Preprint.

Press, S. James (1972), *Applied Multivariate Analysis*, Holt, Rinehart and Winston, N. Y.

Samorodnitsky, G. and Taqqu, M. S. (1992), *Stable Non-Gaussian Random Processes*, Preprint.

Self-Validating Computations of Bivariate Normal Cumulative Distribution Functions

Morgan C. Wang
Department of Statistics
University of Central Florida

Abstract

Self-validating computations based upon interval arithmetic can produce computed values having a guaranteed error bound. Conventional algorithms based upon "point" arithmetic, on the other hand, can lead to exceedingly poor results. This paper gives methods for obtaining self-validating results when computing probabilities, illustrated with the bivariate normal cumulative distribution function. The results from this study provide assessments of the accuracy studies of classical "point" algorithms.

1. Introduction

Bivariate normal distribution is one of the most popular bivariate distributions. The computation of the cumulative distribution function (CDF) values of this probability has been of interest to statisticians for many years. Many conventional "point" algorithms such as Owen (1956), Drezner (1978), Divig (1979), and Drezner and Wesolowsky (1990) have been published to compute these probabilities. Some of the algorithms pose accuracy problems, for example, Monahan (1990a and 1990b) discovered that the Drezner's algorithm (Drezner 1978) can produce reasonable results for the first quadrant and positive correlation coefficients but delivers doubtful results for the other cases. Therefore, to develop a self-validating algorithm to compute the CDF values can provide a solid basis for studying the accuracy of these competing algorithms.

Self-validating numerical method is sometimes called automatic error analysis, and it can be achieved in many different ways. We will use interval arithmetic to accomplish the goal of self-validation. This means that we compute an interval which is guaranteed to contain the theoretically correct CDF value. Then the midpoint of this computed interval is the "point" approximation and the half-width of this interval is the guaranteed absolute error bound giving validity to this midpoint approximation. Since we strive to obtain intervals having very small width (less than 10^{-14}), the approximations obtained provide essentially correct values to be used as a basis for comparing the accuracy of outputs from competing point algorithms.

Basic elements of interval arithmetic and a reference list can be found in Kennedy (1990). Algebraic properties of interval arithmetic are given in Moore (1979) and Ratschek and Rokne (1984). Computations involving intervals do, at first sight, seem to be a complicated and inconvenient process. In fact, this is not the case given today's computer hardware which includes standard floating point support. A few simple functions give interval arithmetic capabilities. The implementation of interval arithmetic, and computation of interval inclusions for various functions are described in Wang and Kennedy (1990 and to appear). This paper omits the details of these issues.

In the next section, we will describe a self-validating numerical method for obtaining interval inclusion of the bivariate normal CDF. Then some computed results will be presented in the third section.

2. Self-validating Numerical Method of Evaluating the Bivariate Normal Integrals

Given a random vector $z = (z_1, z_2)'$ having the bivariate normal distribution with mean vector 0, unit variance, and correlation ρ , the probability P of z_1 less than h and z_2 less than k can be expressed as

$$P(h, k; \rho) = \int_{-\infty}^h \int_{-\infty}^k f(z_1, z_2) dz_1 dz_2 \quad (2.1)$$

Let

$$kw(a, b, c, \rho) = \int_a^b \int_{-\infty}^c f(z_1, z_2) dz_1 dz_2$$

It is easy to verify $kw(a, b, c, \rho) = kw(-b, -a, c, -\rho)$. Using this equation along with the well known relationship

$P(0, 0, \rho) = 0.25 + \frac{\sin^{-1} \rho}{2\pi}$ (Owen 1956), (2.1) can be written as

$$\begin{aligned} P(h, k; \rho) = & 0.25 + \frac{\sin^{-1} \rho}{2\pi} \\ & + \text{sign}(h) kw(0, |h|, k, \text{sign}(h)\rho) \\ & + \text{sign}(k) kw(0, |k|, 0, \text{sign}(k)\rho) \end{aligned} \quad (2.2)$$

Our purpose is to find an interval inclusion (an interval which contains the true value of P) of (2.2), i.e., we need to obtain interval arithmetic. Therefore, we can focus our discussion on obtaining interval inclusion for $kw(a, b, c, \rho)$.

Under the transformation $Y = T^{-1}Z$, where T is the triangular matrix

$$\begin{pmatrix} 1 & 0 \\ \rho & \sqrt{1-\rho^2} \end{pmatrix},$$

we have

$$kw(a, b, c, \rho) = \int_a^b \int_{-\infty}^{c(y)} (2\pi)^{-1} \exp(-1/2(x^2 + y^2)) dx dy \quad (2.3)$$

where $c(y) = (c - \rho y) / \sqrt{1 - \rho^2}$. Let us express $kw(a, b, c, \rho)$ in (2.3) as

$$kw(a, b, c, \rho) = \int_a^b h(y) dy \quad (2.4)$$

where

$$u(y) = -y^2/2$$

$$f(y) = (2\pi)^{-0.5} \exp^{u(y)}$$

$$g(y) = \int_{-\infty}^{c(y)} (2\pi)^{-0.5} \exp(-x^2/2) dx$$

$$h(y) = f(y) * g(y).$$

We denote the Taylor coefficients of function $h(y)$ at $m = (a+b)/2$ as

$$(h(m))_0 = h(m)$$

$$(h(m))_k = \frac{1}{k!} \frac{d^k h(y)}{dy^k} \Big|_{y=m} \quad k = 1, 2, \dots$$

and expand the Taylor polynomial of $h(y)$ at $y = m$. After we integrate this Taylor polynomial term by term, we obtain the integration rule

$$ir_n = 2 \sum_{i \text{ even}}^{n-2} (h(m))_i \frac{(b-m)^{i+1}}{i+1} \quad (2.5)$$

and the error term

$$ie_n = 2(h(e))_n \frac{(b-m)^{n+1}}{n+1} \quad (2.6)$$

of (2.4), where n is a positive even number and e is an unknown number in the interval (a, b) . Let $(H)_n$ be an interval inclusion of $(h)_n$, $X = [a, b]$, $M = [m, m]$, and $D = [b-m, b-m]$. We can form an interval inclusion $Ik_n = Ir_n + Ie_n$ of $kw()$ for each even number n , where

$$Ir_n = 2 \sum_{i \text{ even}}^{n-2} (H(M))_i \frac{D^{i+1}}{n+1} \quad (2.7)$$

is an interval inclusion of (2.5) and

$$ie_n = 2(H(X))_n \frac{D^{n+1}}{n+1} \quad (2.8)$$

is an interval inclusion of (2.6). Now, let

$$KW_0 = Ik_0$$

$$KW_n = KW_{n-2} \cap IK_n \quad n = 2, 4, \dots$$

Clearly, $kw()$ contains in KW_n for each n and $\{KW_n: n = 0, 2, 4, \dots\}$ is a nested sequence. Therefore, we can obtain a resulting interval inclusion of (2.2) iteratively. And this resulting interval can satisfy any specified accuracy requirement.

The only remaining difficulty is to compute these Taylor coefficients in the integration rule and error term. Fortunately, the automatic differentiation techniques can be applied to overcome this difficulty. Detailed descriptions of automatic differentiation can be found in Moore (1979), Rall (1981), and Corliss (1988). Implementation of these automatic differentiation using interval arithmetic provides the means for computing interval inclusion of the Taylor coefficients over the interval M or X . We will not give the details of these computations because they have been provided in Wang and Kennedy (1990 and to appear).

3. Conclusions

Three experiments were constructed to test the performance of this self-validating algorithm for different ρ values. In all the integrals evaluated in each of these experiments, the width of the interval inclusion of the computed probability was not larger than 10^{-16} , so the middle point of computed interval inclusion was necessarily very close to the theoretically correct probability. And the half-width of the computed interval is the maximum absolute error bound of the middle point approximation.

The results from the first two experiment are not present. The third experiment includes twelve integrals with very large correlation. Table I gives the necessary description of these integrals and the computed interval inclusions.

Table I
Interval Inclusion of Integrals Used for the Third Experiment

I.D.	HH	KK	RHO	Inclusion of Probability	
				Lowerbound	Upperbound
1	0.0000	0.0000	-0.9999	0.0022508095471555	0.0022508095476480
2	0.1000	0.0000	0.9999	0.4999999999999994	0.4999999999999995
3	0.1250	0.0000	0.9999	0.4999999999999999	0.5000000000000000
4	4.0000	0.0000	-0.9999	0.4999683287581668	0.4999683287581669
5	0.0000	4.0000	-0.9999	0.4999683287581668	0.4999683287581669
6	8.0000	8.0000	0.9999	0.9999999999999993	0.9999999999999994
7	7.0000	9.0000	-0.9999	0.9999999999987201	0.9999999999987202
8	-3.875	7.6250	-0.9999	0.0000533123497388	0.0000533123497389
9	-5.000	5.0000	0.9999	0.0000002866515718	0.0000002866515619
10	-0.0125	-0.00675	-0.9999	0.0002252159041540	0.0002252159041541
11	-2.500	-3.7500	0.9999	0.0000884172852008	0.0000884172852009
12	5.0000	5.0000	0.9999	0.0000002866515618	0.0000002866515619

Interval arithmetic and automatic differentiation were used to compute interval inclusions of desired probabilities. The length of computed intervals were made sufficiently small so that the probabilities guaranteed to be correct essentially to machine precision were obtained. The cost of our self-validating algorithm is that it takes about 10 times as much CPU time as conventional point implementation algorithm.

4. References

- Corliss, G. F. (1988), *Application of Differentiation Arithmetic, Reliability in Computing: The Role of Interval Methods in Scientific Computing*, New York: Academic Press.
- Divig, D. R. (1979), Calculation of Univariate and Bivariate Normal Probability Functions, *Annals of Statistics*, 7, 903-910.
- Drezner, Z. (1978), Computation of Bivariate Normal Integral, *MATH. COMP.*, 32, 277-279.
- Drezner, Z. and G. O. Wesolowsky (1990), On the Computation of the Bivariate Normal Integral, *J. Statist. Comput. Simul.*, 35, 101-107.
- Kennedy, W. J. (1990), Statistical Computing Feature: Special Purpose Numerical Tools for Approximating Functions, *Statistical Computing and Statistical Graphics Newsletter*, 1, 3-6.
- Monahan, J. (1990a), Quips and Queries, *Statistical Computing and Statistical Graphics Newsletter*, 1, 16-17.
- Monahan, J. (1990b), Quips and Queries, *Statistical Computing and Statistical Graphics Newsletter*, 2, 13-14.
- Owen, D. B. (1956), Tables for Computing Bivariate Normal Probabilities, *Annals of Mathematical Statistics*, 27, 1075-1090.
- Rall, T. B. (1981), *Automatic Differentiation: Techniques and Applications*, Lecture Notes in Computer Science, 120, New York: Springer Verlag.
- Wang, M. and W. J. Kennedy (1990), Comparison of Algorithms for Bivariate Normal Probability Over a Rectangle Based on Self-Validated Results from Interval Analysis, *J. Statist. Comput. Simul.*, 37, 13-25.
- Wang, M. and W. J. Kennedy (to appear), A Numerical Method for Accurately Approximating Multivariate Normal Probabilities, *Computation Statistics and Data Analysis*.

Computation of the Multivariate Hypergeometric Distribution

Trong Wu
Department of Computer Science
Southern Illinois University at Edwardsville
Edwardsville, Illinois 62026

Abstract

The *multivariate hypergeometric distribution* is a natural extension of the *hypergeometric distribution*. The computation of the multivariate hypergeometric distribution is of interest to many researchers who are working in the computing sciences and related disciplines. Currently, there are no software, algorithms, or tables available for computation or reference. This paper presents an effective method to compute the multivariate hypergeometric probability function accurately and efficiently. The method applies prime number factorization to all of the factorials, and cancels all the common factors of the numerator and denominator to reduce the computational complexity to a minimum. We use the Ada programming language for this computation instead the traditional FORTRAN, because the predefined features in the Ada language are suitable for this type computation. This computation can be done currently available machines and time required for the computation is reasonably small.

1. Introduction

Consider a finite population of M objects, of which m_1 are of type 1, m_2 of type 2, ..., m_k of type k , with $m_1 + m_2 + \dots + m_k = M$. Suppose a sample of size N is chosen, without replacement, from among these M objects. Then the joint distribution of the random variables n_1, n_2, \dots, n_k representing the numbers of objects of types 1, 2, ..., k respectively in the sample, is defined by

$$h(n_1, n_2, \dots, n_k; m_1, m_2, \dots, m_k) = \left(\prod_{i=1}^k \binom{m_i}{n_i} \right) / \binom{M}{N}, \quad (1.1)$$

with $n_1 + n_2 + \dots + n_k = N$; $0 \leq n_i \leq m_i$ for $i = 1, 2, \dots, k$.

This distribution is called the *multivariate hypergeometric distribution* with parameters N, m_1, m_2, \dots, m_k [6, 11]. Actually there are only $(k - 1)$ distinct variables, since

$$n_k = N - (n_1 + n_2 + \dots + n_{k-1}).$$

When $k=2$ it reduces to the ordinary hypergeometric distribution. In this special case, we may think that objects can be classified according to some property into n of one group and $N - n$ of another. For example, we might classify a large number of manufactured products as *defective* or *non-defective*. This special discrete function is frequently called the *hypergeometric probability function* [2, 11], because the values $h(x; r, n, N)$ can be expressed as successive terms of a Gauss hypergeometric series.

In many cases, accurate probabilities are very important to the application. Today, the computation of the multivariate hypergeometric probability function is still difficult due to the limitation of the computer systems such as overflow, underflow, and maximum accuracy. Inaccurate results are caused by rounding errors that are induced by many redundant computations in multiplications and divisions. As a result, there are no software packages currently available for the computation of this function such as the IMSL Library [5], minitab [8] and other software packages; in fact, currently there is no effective algorithm available for dealing with computations. Indeed, we need an efficient algorithm and better programming techniques to write a reliable program to accomplish this computation within a reasonable amount time. This paper presents an effective method to compute the

multivariate hypergeometric function accurately and efficiently. Section 2 presents mathematical foundations for solving this problem. Section 3 develops an effective algorithm for this computation. Some computational examples are given in Section 4, and finally, conclusions are given.

2. The Mathematical Foundations

Among the problems of the computation of the multivariate hypergeometric probability function are the computation of factorials, eliminating all the redundant computations, handling multiplications, and minimizing the divisions without overflow and underflow. Methods of managing the computation of factorials and to eliminating all the redundant computation belongs to mathematics while the ways of dealing with multiplications and divisions are in the programming domain. These are two separate issues. First to reduce the computational complexity, we need theorems from the theory of numbers [3] which are stated and proved as below:

Theorem 1. Let p be a prime. Then the exact exponents of p that divides $n!$ is

$$\left\lfloor \frac{n}{p} \right\rfloor + \left\lfloor \frac{n}{p^2} \right\rfloor + \left\lfloor \frac{n}{p^3} \right\rfloor + \dots,$$

where $[x]$ is the largest integer less than or equal to x .

Proof: For

$$\begin{aligned} n! &= 1 \cdot 2 \cdot 3 \cdots (p-1) \\ &\cdot p \cdot (p+1) \cdot (p+2) \cdots 2p \cdots (p-1)p \cdots \\ &\cdot p^2 \cdot (p^2+1) \cdot (p^2+2) \cdots \\ &\cdot p^3 \cdot (p^3+1) \cdot (p^3+2) \cdots \\ &\cdots \cdots (n-1) \cdot n. \end{aligned}$$

We see that the number of p 's factors is $[n/p]$, the number of p^2 's factors is $[n/p^2]$, the number of p^3 's factors is $[n/p^3]$, and so forth. Then the Theorem follows.

From Theorem 1, we are able to factor the $n!$, for all $n \geq 1$, as a product of prime numbers. The result is given in Theorem 2 below:

Theorem 2. For any positive integer $n \geq 2$, the $n!$ can be written as a product of prime numbers.

$$n! = p_1^{r_1} \cdot p_2^{r_2} \cdot p_3^{r_3} \cdots p_k^{r_k}, \quad (2.1)$$

for some positive integer k .

Example 1: Consider $20!$, we have the following exponents of prime numbers:

$$\begin{aligned} \text{The exponent of } 2 \text{ is } \left\lfloor \frac{20}{2} \right\rfloor + \left\lfloor \frac{20}{2^2} \right\rfloor + \left\lfloor \frac{20}{2^3} \right\rfloor + \left\lfloor \frac{20}{2^4} \right\rfloor \\ = 10 + 5 + 2 + 1 = 18. \end{aligned}$$

$$\text{The exponent of } 3 \text{ is } \left\lfloor \frac{20}{3} \right\rfloor + \left\lfloor \frac{20}{3^2} \right\rfloor = 6 + 2 = 8.$$

$$\text{The exponent of } 5 \text{ is } \left\lfloor \frac{20}{5} \right\rfloor = 4.$$

$$\text{The exponent of } 7 \text{ is } \left\lfloor \frac{20}{7} \right\rfloor = 2.$$

The exponents of 11, 13, 17, and 19 are all equal to 1. Hence,

$$20! = 2^{18} \cdot 3^8 \cdot 5^4 \cdot 7^2 \cdot 11 \cdot 13 \cdot 17 \cdot 19.$$

3. The New Algorithm

For simplicity, we may simplify the equation (1.1) into a division of two products of factorials. Hence, we have

$$\begin{aligned} h(n_1, n_2, \dots, n_k; m_1, m_2, \dots, m_k) \\ = \left(\prod_{i=1}^k \binom{m_i}{n_i} \right) / \binom{M}{N} \\ = \frac{\binom{m_1}{n_1} \binom{m_2}{n_2} \cdots \binom{m_k}{n_k}}{\binom{M}{N}} \end{aligned}$$

$$= \frac{\prod_{i=1}^k m_i! \cdot N! \cdot (M-N)!}{M! \cdot \prod_{i=1}^k n_i! \cdot \prod_{i=1}^k (m_i - n_i)!}, \quad (3.1)$$

Now, we are able to develop a computational algorithm for this distribution:

Algorithm 1.

(1) Apply the prime number factorization, given in identity (2.1), to all the factorials, $N!$, $M!$, $(M-N)!$, $m_i!$, $n_i!$, $(m_i - n_i)!$ ($i = 1, 2, \dots, k$) in equation (3.1).

(2) Cancel the common factors in the denominator and numerator. Obtain an irreducible fraction for computation (see Example 2).

Note that the computation in (1) requires the use of a sequence of prime numbers that can be computed either by Eratosthenes' sieve algorithm or by an improved algorithm given by Luo [7]. After (2), all the common factors in the numerator and denominator are cancelled and reduced to an irreducible form; the number of multiplications and divisions is minimized.

Example 2. A petroleum corporation has 50 gasoline stations in a certain state; it has classified them according to merit of geographic location as follows:

Location Excellent Good Fair Poor Disastrous

Stations 10 12 8 15 5

The corporation has a computer program for drawing random samples, without replacement, of its stations. The joint probability of obtaining a sample of 20 of these stations with 2 excellent, 4 good, 1 fair, 8 poor, and 5 disastrous is given

$h(2, 4, 1, 8, 5; 10, 12, 8, 15, 5)$

$$= \frac{\binom{10}{2} \binom{12}{4} \binom{8}{1} \binom{15}{8} \binom{5}{5}}{\binom{50}{20}}$$

$$= \frac{10! 12! 8! 15! 5! 20! 30!}{2! 8! 4! 8! 1! 7! 8! 7! 5! 0! 50!}$$

After factorization and cancellation, the above expression is reduced to its simplest form.

$$= \frac{3^6 \cdot 5^2 \cdot 11 \cdot 13}{7^2 \cdot 23 \cdot 31 \cdot 37 \cdot 41 \cdot 43 \cdot 47}$$

$$= 2.433134239681592191426386865530844E-5$$

4. Some Computational Results

We use the Ada programming language [1] to implement the algorithms given in the Section 3 because the language is an ANSI standard language; it has special features called *exception handling* and *tasks*. These features not only make programming simple, but also prevent a program crash due to overflow or underflow. In the actual implementation, we use three *tasks*; *task one* and *task two* are employed to perform the multiplications in the numerator and denominator respectively; *task three* is used to perform a division. When both products in the numerator and denominator reach a maximum, both *task one* and *task two* stop temporarily and invoke *task three* to perform a division of the products that have been obtained in the numerator and denominator before an overflow occurs. After *task three* completes its job, *task one* and *task two* resume their computation and repeat this procedure until the final result is obtained. These *tasks* work together and guarantee that the result of this computation will be the most accurate.

Some machines allow users to have a precision of 33 significant digits or a 128 bit floating point number [10, 12]. For the 128 bits, 1 is used as a sign bit and 15 are used as the exponent field; the remaining 112 bits together with a hidden bit gives 113 significant bits or 33 significant digits, according to the IEEE standard 754 floating-point number format [4, 9]. The programs ran on a MicroVax II machine with the VMS V.4 operating system and the Vax Ada V.1 compiler.

The following sample results were obtained from an output of an Ada program; this program was running on a MicroVax II machine. The program can compute the probabilities of the multivariate hypergeometric distribution with an arbitrary number of events and no limitation on the number of occurrences of each event. Some results and their computation times are given as follows:

- (1) $h(2,3,5,0; 5,6,7,2)$
 $=2.273268527138496178743856762432613E-2$
 (0.25 seconds)
- (2) $h(2,4,1,8,5; 10,12,8,15,5)$
 $=2.433134239681592191426386865530844E-5$
 (1.13 seconds)
- (3) $h(5,8,12,2,7,10,12,15; 5,20,18,3,9,10,16,21)$
 $=1.814285332418592083659890283423809E-7$
 (1.66 seconds)
- (4) $h(7,13,23,5,11,37,47,20,15,50; 10,20,30,40,50,60,70,80,90,100)$
 $=8.025173894645222350927412268425642E-33$
 (7.95 seconds)

5. Conclusions

The computation of the multivariate hypergeometric distribution is in general a critical problem due to the limitations of computer systems and programming techniques. The goal of a computation is accuracy; the time consumed for the computation must also remain reasonably small. This research has developed a method based on theorems from the theory of numbers and implemented them in the Ada programming language; the former is to overcome the limitations of a computer system, and the latter is to solve the technical difficulty in programming. The method is machine independent; precision is arbitrary, subject to storage limitation. The Ada language is available for most supercomputers, mainframes, medium sized computers, and personal computers. Since this computation is a number theory problem in nature, the problem can only be solved with number theory. The results given in Section 4 have reached the predefined goal.

References

1. Barnes, J.G.P. *Programming in Ada*, Third Edition, Addison Wesley Publishing, Reading, Mass., 1989.
2. Gross, A. J. and Clark V. A. *Survival Distributions: Reliability Applications in the Biomedical Sciences*, John Wiley & Sons, New York, 1975.
3. Hua, L. K. *Introduction to Number Theory* (English Translation), Springer-Verlag, New York, 1982.
4. IEEE Inc. *IEEE Standard for Binary Floating-Point Arithmetic (ANSI/IEEE std 754-1985)*, New York, 1985.
5. IMSL Library *FORTTRAN Subroutines for Mathematics and Statistics, User's Manual*, IMSL, Inc., Houston, Texas, 1984.
6. Johnson, N. I. and Kotz, S. *Discrete distributions*, Houghton Mifflin Company, Boston, Mass., 1969.
7. Lou, X. "A practical sieve algorithm for finding prime numbers," *Communications of ACM*, 32, 344-346, 1988.
8. Ryan, T. A., Joiner, B. L., and Ryan, B. F. *Minitab, Student Handbook*, Duxbury Press, North Scituate, Mass., 1976.
9. Stevenson, D. *A proposed Standard for Binary Floating-Point Arithmetic*, *IEEE Computer*, 14, 51-62, 1981.
10. Struble, G. *Assembler Language Programming, The IBM System/370 Family*, Addison-Wesley Publishing Reading, Mass., 1984.
11. Trivedi, K. S. *Probability & Statistics with Reliability, Queuing, and Computer Science Applications*, Prentice-Hall, Inc., Englewood Cliffs, New Jersey, 1982.
12. *Vax Ada Language Reference manual*, Digital Equipment Corporation, Maynard, Mass., 1985.

**IMPROVING THE JUDGMENT OF PROFESSIONAL SPORTS MANAGERS
BY USING SYSTEMATIC OBSERVATION OF A REAL-TIME
PROCESS AND CREATING BETTER 'GAME' STATISTICS
(The case of NHL hockey)**

Pierre A. Balthazard
The University of Arizona
Tucson, AZ 85721

Kevin J. Leonard
Wilfrid Laurier University
Waterloo, Ontario

INTRODUCTION

This paper presents the framework for the use of a historical database structure to support evaluative and predictive judgement in the domain of a major professional sports league, namely the National Hockey League. On the platform of a Relational Database Management System, value is added to game statistics by keeping true, non-character literal, temporal attributes and interfacing the resulting temporal database management system to a video representation of a real-time process. This framework allows for the link between image and textual data in a direct access mode through the compilation of complex regular and temporal queries.

The paper addresses the physical problem of efficiently recording, storing, and retrieving large amounts of match related data to create an accurate and complete description of performance. This information is needed to form the basis for proper judgement by the NHL decision maker.

APPLICATION BACKGROUND

We have chosen the domain of professional sports. Although having high profiles, sports teams can be best described as medium sized companies, usually having less than 100 employees, often with revenues not reaching \$100 million per year. More specifically, we have implemented a database solution to the planning and control problems of the real-time process of playing a professional hockey game. The game of hockey is an interesting research domain. In a competitive life span of 60 minutes, a finite set of events can occur which result in success or failure. Unlike the longer time span of the "going concern", in hockey one gets instant gratification for successful strategies and quick feedback on failures. Our current goal is

to offer the NHL decision maker a means to enhance team preparation for future performance through the systematic and complete study of past performances. This is accomplished by adding meaning to information previously unused or unknown to the decision maker. Our long-term research goal is to define the parameters and implement expert systems that will support decision making for real-time "stochastic" processes.

Our research has found that current team preparation revolves around subjective evaluation of performance and review of game videos. Data capture is primarily limited to box scores and discrete events such as goals and scoring chances. These videotaped hockey games through repeated and systematic visioning, have become de facto executive support systems for the NHL decision maker. We argue that a hockey game is made up of more than discrete events and that these events are all related. Hence, value can be added to this information and better data can be captured by a more effective and efficient use of technology.

The manner in which information is categorized during a real-time process is a major problem. Firstly, there is a problem in deciding what information is relevant and needs to be captured (section 1). Secondly, there is the physical problem of efficiently recording, storing, and retrieving large amounts of data (section 2). This paper discusses solutions for data capture and improved systematic observation criteria (section 3) and proposes a semantically-improved database framework that can best represent this and other real-time processes (section 4).

1. CATEGORIZATION OF INFORMATION

1.1 Saving All the Bits

In a parallel discussion to that of this paper, Denning (1990) discusses that it is important to "save all the bits"

generated by an instrument or computation. He argues firstly, that the cost of acquiring the bits is so great that we cannot afford to lose any of them and, secondly, that some rare event might be recorded in those bits, and to throw them away would be a great loss for the advancement of knowledge in the domain.

To counter-balance this paradigm, one must remember the impossibility of doing this in practical terms. The rate and volume of information in many real-time applications can overwhelm our networks, storage devices, and computers, as well as (and more importantly) the human capacity for comprehension. Humans have limited information-processing capabilities (Hogarth, 1980). These involve selective perception, sequential processing, limited processing capacity, limited memory capacity, memory by association, reconstruction of memory, etc.

The goal is to complement the human processing limitations by extending it using computer technology without overpowering the technology. This delicate balance is often the difference between quality information and data explosion.

1.2 Systematic Observation

The concept of systematic observation was developed in the field of education and was introduced to collect data on teachers in a classroom environment. Systematic observation instruments consist of a number of predetermined, clearly defined categories of behavior, as well as definite rules and procedures for their identification and coding. The focus of the instrument is directed toward a critical element in the learning process and is based upon a sound theoretical framework. The usual procedure is for the recorder to observe either the teacher or the student during a learning situation and code the targeted behaviors in accordance with the procedural rules of the instrument. These systematic observation techniques provide a method of obtaining objective, reliable, and valid measurements of behavior.

We feel that there is an intimate link between the process described above and that of capturing the events that occur during real-time processes. In the next section, we define the "predetermined, clearly defined categories of behavior", as well as the "definite rules and procedures for their identification and coding" of performance during a hockey game.

1.3 Saving the Appropriate Bits

We have followed an NHL head coach for over two years and have held regular question and answer sessions with the team's coaching staff. We conclude that hockey

performance models are based on the perception that past behavior is often the best predictor of future behavior and that through the study of past behavior one can develop specialized training, teaching schedules, and game strategies.

The study of past behavior is done by the review of past game videotapes. Some NHL hockey teams have incorporated video editing software as "systematic observation" tools. These video editing packages allow a team to stamp events while they are watching a videotape of a game and return to these clips at their convenience. Our coach's systematic observation of performance model is currently based on the documentation of seven types of events: scoring chances for, scoring chances against, body checks, face-offs, minutes played, player rating, and key game identification fields. All items are discrete on-ice action except "player rating" which is a subjective after-the-fact evaluation and "key game identification fields," which are key attributes to a hockey game. This setup uses a "flat files" approach where queries are usually restricted to within-event type information.

2. RECORDING, STORING, and RETRIEVING

Although rudimentary in nature, the system based on the current performance model does produce a complete statistics report that cross-references the 7 "flat files" data from many angles. The quality of this recurring batch report has received international attention (Sexton 1990).

The current formula for systematic observation solely stores the on-ice game statistics (primarily data on scoring chances, face-offs, and minutes played). Benefits of automating this process includes ad hoc query potential and the tabulation of numbers for statistical inference. At this time, there is no link between the video editing software and the statistical "flat files" system. Coaches must first watch the game and edit out highlights with the editing software. Once this process is complete, they then enter the statistical package and manually re-enter all the highlights from a statistical perspective -- who was involved, the opponent, and other pertinent information. Statistical analyses are then performed on these data and reports are generated. This is all done separate from the video editing software.

3. IMPROVED IDENTIFICATION AND CODING

The coach's current model has three major weaknesses: Firstly, it only addresses a subset of possible occurrences in a hockey game; secondly, it does not offer a natural connection or sequence of events between the flat files; and thirdly, it does not allow for temporal attributes.

These weaknesses are however not inherent to this application but to all real-time processes. Often a continuous process is simplified to a set of discrete processes which are often more practical to work with. For example, taking a patient's temperature twice a day and defining between readings mid-points by linear interpolation represents continuous change as depicted by a finite set of measurements. Continuous processes are often translated to a discrete representations in order usually to better operationalize their capture and retrieval in automated databases.

But in other circumstances, strategic advantage is created by building systems that can represent continuous processes as such. The banking industry capitalizes on this point by conducting business transactions according to the generally accepted daily cycle although its internal technology updates credits in real-time.

Through a maturing process, today's systems must satisfy more sophisticated requirements for information. Many articles (Cash and McLeod 1985; Parsons 1983) have focused attention on how developments in information systems technology have made many new applications that have strategic importance feasible. Cash and McLeod argue that only systems that add value to information will be used in *winner* companies. Systems must leave the "horseless carriage" epoch of automation into what Zuboff describes as *informating* (Zuboff 1988).

The remainder of this paper outlines a method for the systematic observation of real-time data in real-time through the use of recently available technology and a generic but exhaustive list of behaviors, rules and procedures of NHL hockey.

3.1 Performance Behaviors of NHL Hockey

We have compiled a list of 24 "recordable" game occurrences (13 real-time events, 8 stopped time events, and 3 identification records). We feel that this is an exhaustive list of events in a hockey game. The use of pen-based technology will allow for real-time input of all this information and more with the processor controlled implicit time-stamping of events. Currently being investigated at the University of Arizona is a character recognizing writing pad and template developed by a multi-national high-technology corporation (Briggs 1990). In this new peripheral, a small radio-signal emitting device is embedded in a stylus (special pen) and detected by a rectangular (8 X 12) writing pad. The pad senses and recognizes movements that form characters or commands. It is used in lieu of the keyboard.

3.2 Operationalizing Systematic Observation

A typical hockey play can be captured fully with as little as 1 or 2 *pen* strokes or at most (full line changes - both teams) approximately 12. We have estimated that the most complex hockey play can be captured in less than 3 seconds for the relatively new user. We are now conducting experiments based on the power law of practice to estimate capture time for experts. Each registered play becomes an object occurrence. Data capture will be a two-pass process: in real-time mode and review mode. Sequence of events will be captured in real-time. They will be edited and further described upon review.

Templates for various single-user and networked DOS based systems have been developed and tested, and a series of experiments on their usage conducted. Preliminary results show that top business executives, given 30 minutes to learn the technology, have character recognition rates surpassing 90%. These experiments and others (Mahach 1989) have also shown that, in circumstances where input of data is made up of a small amount of keystrokes, the writing pad is preferred over traditional keyboards.

4. SEMANTICALLY-IMPROVED DBMS STRUCTURE

4.1 Time and Databases

Research and development over the last twenty years has culminated in the widespread use of database management systems. As usage has grown, the desire to capture more data semantics has led to the development of data models that provide the concepts and corresponding formalism in which an image of the real world can be expressed. Classic models do not represent time in a natural, real-world way. Time is not merely another dimension, or another data item tagged along with each tuple, but rather a more fundamental organizing aspect that human users treat in very special ways. In current DBMS time is either neglected, treated implicitly, or explicitly factored out. Most data models presume the database to represent a single current status (Tsichritzis and Lochovsky 1982). They provide only state transitions. However, Schueler noted in 1977 that even simple updates to databases destroys valuable information (Schueler 1977).

In many cases information seekers are not only interested in a representation of a current state, but also in the history of earlier states or a prognosis of future states. Historically it has been difficult to reconcile infinite conceptual models with finite machines and finite memories. Through dramatic increases in processing power and

the introduction of large-capacity directly accessible secondary storage devices, the tools now exist to implement a system that truly represents time.

A widely used DBMS time concept is the *event*. Traditionally, time stamping of events is accomplished using character literals. An event is a change of status, which is kept until the next event. Events can occur at integral time points (Breutman 1979), or at real time points (Bubenko 1977). In other cases, such as the capture of hockey information which has a pattern of continuous change and some fragmentary observations, the notion of event is not a good modeling approach. A hockey game has a *life* of 60 timed minutes. In this relatively short time-span, many hundred (often overlapping) events occur: plays, line combinations, game situations. A traditional database using only a character literal time stamped representation of events does not offer the time granularity desired to properly record a hockey game. An accurate representation of systematic observation model must include events in the traditional sense, time intervals, pre-events -- behaviors or performance prior to an event, and post-events -- behaviors or performances after an event.

By recording a sequence of time stamped object occurrences in a suitable DBMS, new types of entities are then created: *intervals* are a structure with a time interpolation that includes all action between two time boundaries; *pre-events* are actions that lead to an event; and *post-events* are actions that occur after an event. In hockey, one might want to see the play leading to a goal or one might want to know what happened in the 30 seconds after a penalty was called.

This is why the use of a temporal database is recommended. Instead of events, games can be transcribed as *intervals* of action. A semantically well-grounded temporal database will allow for second by second granularity, thus precise intervals that can capture the essence of hockey.

4.2 Temporal Databases

When discussing temporal databases (TDB's), a review of the taxonomy is necessary. Conventional databases model an enterprise using *snapshots* (Bubenko 1977). A snapshot is a state or an instance of a DBMS with its current content, which may or may not represent a current status (Ahn 1986). A *rollback* database resolves the snapshot DBMS problems by recording a sequence of past states, indexed by time (Ariav 1986; Gadia 1988). This approach requires a representation of transaction time -- a time stamp representing the time of day when the transaction occurred. In a rollback DBMS changes can be made only to the latest state. *Historical* DBMS record a single historical state per relation. Changes are made and

a new version of the database created. In historical DBMS previous states are not retained but there is support for valid time -- a second time stamp that depicts the actual time an entity is active and not when it was updated (Laning 1982; Snodgrass 1987). A *temporal* DBMS supports both transaction (when action is recorded) and valid (when action actually occurred) time in the same relation. A temporal database allows for a relative (virtual) time frame instead of always dealing with the present (Snodgrass 1987).

In hockey, time has two dimensions: Firstly, a sequence of object occurrences in a hockey game and, secondly, a sequence of completed hockey games. Each object occurrence is time stamped using two different type of time attributes: "hockey time" splits into 3 periods of 20 minutes each and assures the capture of data sequence, and "time of day" which captures image sequence and is used for synchronization of data with the videotape version of a match which is often taken directly from a network "feed" (commercials and all). These two time attributes work in parallel and are on the same plane. The second time dimension is required to denote the sequence of games. A regular NHL season has 80 games per team.

4.3 Schematic Structure of Hockey TDB

The external user's view is that of building blocks, each block having the dimensions of time, objects, and attributes. Individual blocks store the information for a given game and a regular season would combine 80 such blocks.

Two time attributes are used in this framework: actual clock time and game time. Clock time is used to access the videotaped images of the game. This will allow for a more flexible use of the videos since it eliminates the use of video time codes. Video time codes are electric pulses on videotape that, when applied to a frame by assigning it a unique frame number, help a technician edit the images very precisely. In this application, the access to the video is controlled by the database management system and not the video editing software. This approach will however not allow for frame accurate video editing but does offer a reliable and powerful, second by second, granularity of image sequence.

IMPROVING JUDGEMENT

The proposed system is both a decision support and an executive support system. A common and accepted distinction between an ESS and a DSS is based on their respective applications: The former, like executives, deals with unstructured and ambiguous information and circum-

stances; the latter is intended for more structured, repeated, quantitative model-based decisions. In hockey, goals and assists, time on the ice, plus and minus statistics, and a multitude of other generated statistics are repeated and analyzed. Executive support systems do not easily lend themselves to the sorts of explicit quantitative models a typical DSS can provide. Instead, the coach must rely on his own implicit performance model to manipulate and interpret this information. For example, coaches often try to find intangible advantages in non-statistic, often video-based, information.

Much of our work is thus appropriately addressed to designing a tool that can not only support the repetitive statistics but also the performance model. A successful system must support the coach's cognitive processes of how to make decisions and forecasts. Already accomplished are the development of a relationship with a sports team and a detailed investigation of their perceived data requirements. This has lead us to define a relational model for data and generic model for the systematic observation of performance by linking the data to a video representation.

Our research agenda includes: Firstly, the development of a computerized link whereby both data entry and analysis combines image and textual data. Secondly, modifying the input and edit device from the standard keyboard to the stylus/tablet. This will result in a more natural interface for the hockey executive. Thirdly, completing the model by capturing more of the available data and incorporating temporal data to better reflect time relationships (fatigue factors, dependent events, momentum swings) that occur in professional sports. And subsequently, applying available technologies (videodiscs, CD-ROM, multi-media screens, workstation processors) to enhance the reviewing and analysis of game highlights to be performed using a shared textual/video screen.

References

- Ahn, I. "Towards an Implementation of Database Management Systems with Temporal Support," *Proceedings of the international Conference on Data Engineering (Los Angeles)*, IEEE Press, New York, 1986, 374-381.
- Benbasat, I., and Taylor, R.N., "The impact of Cognitive Styles on Information Systems Design," *MIS Quarterly*, Vol. 2, No. 2, June 1978, 43-54.
- Breutman, B., "The Temporal Dimension of Conceptual Schemas," in *Proceedings of IFIP West Germany 2.6*, Munich, March 1979.
- Briggs, R. "Testing Managers' Acceptance of Notebook Technology over Keyboard Technology: An Interim Report," *University of Arizona (MIS) Working Paper Series*, December 1990.
- Bubenko, J.A., "The Temporal Dimension in Information Modeling," *Architecture and models in Data Base Management Systems*, North-Holland, Amsterdam, 1977, 93-118.
- Cash, J.I., and McLeod, P.L., "Managing the Introduction of Information Systems Technology in Strategically Dependent Companies," *Journal of Management Information Systems*, Vol. 1, No. 4, Spring 1985, 9-25.
- Clifford J., and Tansel, A.U., "On an Algebra for Historical Relational Databases: Two Views," *Proceedings of ACM-SIGMOD 1985 International Conference on Management of Data (Austin)*, ACM, New York, 1985, 247-267.
- Denning, P., "Saving All the Bits," in *American Scientist*, Volume 78, Sept-Oct. 1990, pp. 402-405
- Gadia, S.K., "A Homogeneous Relational Model and Query Languages for Temporal Databases," *ACM Transactions on Database Systems*, Vol. 13, No. 4, December 1988, 418-448.
- Hogarth, R., *Judgement and Choice*. John Wiley: Chichester, England, 1980.
- Kuklinski, T.T., "A Case for Digitizer Tablets: The Inherent Advantages are Many," *Computer Graphic World*, May 1985, 45-52.
- Laning, L.J. "A DSS Oversight--Historical Databases," *DSS-82 Transactions*, G.W. Dickson Editor, June 1982, 87-95.
- Lucas, H.C., "Performance and the Use of an Information System," *Management Science*, Vol. 21, No. 8, April 1975, 908-919.
- Mahach, K.R., "A Comparison of Computer Input Devices: Linus Pen, Mouse, Cursor Keys and Keyboard," *Proceedings of the Human Factors Society 33rd Annual Meeting - 1989*, Human Factors Society, 330-333.
- Niles, J.M., Gray, P., Carlson, F.R., and Hayes, J. "The Personal Computer and Society: A Technology Assessment," Presented at the National Computer Conference (Anaheim), June 1978.

Oed, R., and Doster, W., "On-Line Script Recognition - A User Friendly Man Machine Interface," *IEEE Comput* 85, 741-743.

Parsons, G.L., "Information Technology: A New Competitive Weapon," *Sloan Management Review*, Fall 1983, 3-14.

Schueler, B.M., "Update Reconsidered," in *Architecture and Models in DBMS*, G.M. Nijssen editor, North-Holland, Amsterdam, 1977.

Sexton, J. "It Figures, Rangers NHL's Best," New York Times Service, appearing in *The Globe And Mail*, Toronto, November 13, 1990.

Snodgrass, R., "The Temporal Query Language TQuel," *ACM Transactions on Database Systems*, Vol. 12. No. 2, June 1987, 247-298.

Tappert, C.C., Suen, C.Y., and Wakahara, T., "The State of the Art in On-Line Handwriting Recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 12, No. 8, August 1990, 787-808.

Tsichritzis, D.C., and Lochovsky, F.H., *Data Models*, Prentice-Hall, Englewood Cliffs, New Jersey, 1982.

Zuboff, S., *In the Age of The Smart Machine: The Future of Work and Power*, Basic Books, New York, 1988.

Choosing a Computational Environment for Analyzing Large Databases

Paul J. Gregor, Ph.D.

Houston Center for Quality Care and Utilization Studies
Veterans Affairs Medical Center (152)
2002 Holcombe Blvd.
Houston, TX 77030

Abstract: *Recently our small group of researchers was faced with the inadequacy of our computational resources for our ongoing statistical analyses of large datasets. Turnaround times on our network of 386/33 PCs were long, sometimes running 15 to 30 hours. Using a remote large mainframe also had its own problems. We needed a change and a big one, but we didn't have big dollars. This paper describes our short (3 month) exploration of alternatives and our findings.*

We first identified four categories of requirements for our analysis environment. We then looked at available technology for possible solutions, identifying relatively low-cost UNIX RISC workstations as the most-likely candidate platforms for our statistical work. We then devised a way to measure the performance of several different statistical analysis environments with large datasets. Our results were startling: workstations provided close to two orders-of-magnitude improvement in performance at an affordable cost. Our benchmark statistical analysis job which took 3 hours on one of our PCs ran in 12.5 minutes on one workstation and 4.6 minutes on another. Performance for four different types of statistical analysis was found to vary nearly linearly with database size, as size ranged from 1,000 to 1,000,000 records. Hardware costs varied from \$12,000 to \$75,000, depending on the vendor and the configuration options chosen. The newer workstations also supported newer software with a much improved interface and graphics capabilities.

Introduction

This conference was devoted to the "interface between computing science and statistics." Many of the papers have talked about new directions in one or the other of these fields, presenting, for instance, research and prototypes of new data analysis tools and new computational techniques. This paper is a little different. The focus is on the state of the marketplace, not the state of the art or the state of the lab.

This paper reports the results of one group of data analysts searching for adequate tools to look at large data sets. We

were looking for the right "computational environment" - the right combination of hardware, system software, and statistical software (or, perhaps, the right "interface between computing science and statistics") to effectively do our jobs. This paper puts a structure to the process of gathering requirements for such a computational environment. It also presents some of our findings when we matched these requirements to the marketplace. The intention of the paper is not to single out a particular solution for all statisticians, but rather to identify an approach, to point out some things to consider, and to point out some interesting characteristics of "today's" computational environments.

The Setting

Strictly speaking, this paper doesn't report on the market of "today", but rather that of the summer of 1991, which was when we did our analysis. That was when our small group of about a dozen researchers realized we had a problem. One of the goals of our interdisciplinary group (statisticians, programmers, physicians, health services researchers, and a psychologist and economist) is to study and evaluate patterns in health care quality and utilization. We have available to us a number of very large databases describing the operations of the 172 hospitals operated by the US Department of Veterans Affairs. One database, the Patient Treatment File (PTF), contains one 140-variable record for each hospital stay. The PTF has about 1,000,000 records per year, with 11 years of data currently available. A second database describing outpatient care has about 20,000,000 records per year. Our problem was that we felt it was taking an excessive amount of time to make use of these valuable datasets and interpret them.

We formed a small, *ad hoc* committee of folks with diverse backgrounds to come up with a solution. Our meetings seemed to alternate between proposing solutions and identifying requirements. Clearly stated requirements are needed before a good solution can be chosen, but examining candidate solutions can bring out hidden (but real) requirements. The following sections describe some of the requirements and solutions that emerged from our iterations between the two.

Requirements

A basic requirement of our group is to have access to a responsive, easy-to-use, system able to analyze large datasets. The first two requirements (responsive and easy to use) translated into the need for a local system. Accessing a shared facility (such as VA's central data repository) is fine for occasional use of current data from a wide variety of sources. On a routine basis, however, it is awkward, cumbersome and time-consuming, especially when your research analyses have to compete with higher-priority, routine work such as processing payroll and performing financial management.

Initially, our local system was constructed to get away from that problem. The system consisted of a network of 386 PCs connected via Novell software and ethernet cabling to a 486 file server with 5 gigabytes of disk. A tape drive attached to one of the PCs allowed us to import data from VA's central data repository. Our data analysis software was PC SAS, version 6.04. At the time it was purchased (1990), this system represented the most powerful micros available, combined with the most widely used software for sharing data among PCs, and one of the standards in statistical analysis software.

This system worked fine when handling datasets of 100, 1,000, 10,000, or even 30,000 observations. It gave us the descriptive and analytic statistics we needed in a reasonable amount of time - minutes, if not seconds. But routine sorting and analysis of several hundred thousand records took hours. Processing of a million records could take more than 20 or 30 hours. This made it difficult and time-consuming to look at multi-year trends in VA health care utilization.

Thirty hours was clearly unacceptable, but it wasn't at first exactly clear what would be acceptable. After some thought on research staff productivity, we established an objective of increasing throughput from one analysis every day or two (based on 20 to 30 hour response times) to several analyses every working day (hence 2 to 3 hour response times). To have a real effect on productivity we needed an order-of-magnitude improvement in response time; improvements on the order of 50% or 100% would not have gotten us out of the once-a-day cycle.

A second drawback of the system was the statistical software's outdated interface with the user. The origins of SAS go back to the '60s - a time of punched cards, when terminals and interactive system use were rare. The way one used SAS in the '60s was to study the manual, figure out the commands, parameters, and syntax that were

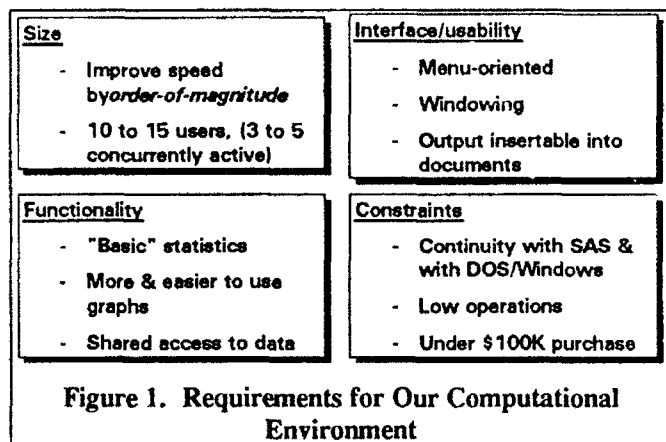
needed, key-punch the statements, submit the cards to SAS, and receive back, hours later, tabular output with crude-looking, line-printer-produced graphs. This is still the basic model for using SAS version 6.04 today! It is awkward, inefficient, dramatically different from interface technology and standards of the '80s, and inconsistent with the '70s and '80s push toward exploratory data analysis.

What we needed was the ability to do truly interactive statistical analysis. At a minimum this entailed being able to easily select analysis procedures, options, and datasets from menus, to change minor characteristics of output without re-running the whole analysis, to quickly see the effects of omitting one or more potential outliers, to see and work with the entire dataset in a tabular format, to look at graphs and the raw dataset simultaneously, to see and work with multiple datasets simultaneously. What we needed was the number crunching of SAS coupled with the user interface of spreadsheets like Excel that run under Windows on PCs. We also thought it would be great if we could easily copy output from our statistical analyses directly into our reports and presentations, again as supported by some word processors and spreadsheets.

Besides improved response time and usability we also tried to assess the functionality that we required. We decided that basically we made use of "standard", proven statistical procedures. We really weren't trying to develop new procedures, and that the statistical functionality of SAS version 6.04 basically covered our needs. We did however feel that we wanted better ways to view and display our data. A key non-statistical function we felt our environment needed to provide was shared access to data. A number of people needed to be able to use our large data sets in parallel - we felt we couldn't support the storage for multiple copies, or the time and effort to move the files among multiple processors.

Our solution space was limited by several constraints. First, we wanted to avoid as much as possible conversion effort and training. Staff had PCs and were using various DOS and Windows tools for preparing reports and presentations. They were used to SAS's command language. We thus sought to provide some continuity with this environment. Second, we are a small group and needed to minimize the overhead of recurring operational support. Though we expected to have a system manager, we sought to control the complexity of the job as much as possible. Finally, our most concrete constraint was our budget. We were told we to keep the solution under \$100,000. When we started our analyses, we really weren't sure whether this would be adequate or not.

Figure 1 summarizes this discussion.



Candidate Solutions

In exploring solutions to our problems we initially considered two traditional solutions: first, upgrading the PCs and/or the network, and second, buying a small local mainframe. The first approach was rejected for several reasons. Based on the general experience of others and reviews in the popular PC magazines, upgrading to 486s at that time would have only achieved at absolute best a doubling of processing capability - nowhere near the needed order-of-magnitude improvement. Besides the limited performance, staying in a DOS environment would have left us with older, inadequate SAS software with a poor user interface. We also discovered SAS 6.04 had significant memory management limitations, making use only of 2 MB of memory even though our processors had more. As for upgrading the network, based on our performance measurements, the network was not the limiting factor anyway. Response time was essentially the same whether the data file resided on the server or local disk.

The second approach, installing a small mainframe locally, was briefly explored and rejected. Major factors in the decision were the high initial purchase cost (at least double the funds available) and the high recurring cost for maintenance and operations.

The approach we settled on was to maintain our existing network and to add to it a computational server which would handle high-speed processing of the large datasets, as conceptualized in Figure 2. The PCs were preserved as-is to handle word processing, spreadsheets, and normal presentation graphics without any changes in procedures. To these functions was to be added the ability to start and interact with analysis software running on the computational server. The file server would continue with the same Novell protocol to handle routine document file sharing and fairly static data files which were not

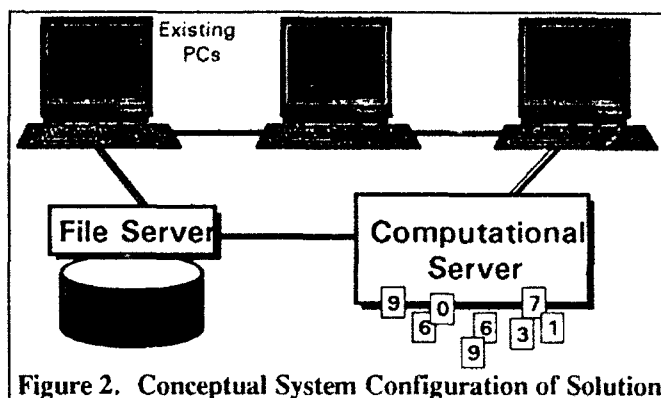


Figure 2. Conceptual System Configuration of Solution

frequently used. The ability to move data files between the computational server and the file server was also to be provided.

The server was to be selected from the relatively large group of RISC computers using the UNIX operating system. These "workstations" have developed a large presence in the engineering community where they are used for handling high-volume engineering and image analyses. Such processors are available from virtually every large computer maker. These processors were relatively low priced - significantly less than mainframes and, in some cases, approaching the cost of large PCs.

A side advantage of a UNIX system was that it would support SAS's latest version (6.07), which had an improved user interface and graphical abilities. A new SAS product (Insight) would provide some of the spreadsheet-like data editing and viewing features we wanted.

A potential complication was the interface between the user at the PC and the statistical procedures executing on the UNIX processor. SAS's user interface in the UNIX environment uses standard windowing capabilities provided by the X-Windows protocol and routines. X-Windows was designed to separate display management from data generation, with each residing on its own processor in a network. This characteristic fit right in with our general concept of using the UNIX box for calculations and the PCs for displays. But X-Windows initially was built for use between two UNIX processors using the TCP/IP protocol for communications across the network. We wanted one of the processors to be DOS/Windows, not UNIX, and our network used Novell protocol, not TCP/IP. We briefly considered switching all the PCs to the UNIX operating system and dropping the Novell protocol, but that would have been a drastic change for our staff and would have locked us out of the popular DOS/Windows standard office software.

Fortunately, computing technology seemed to be evolving along with our requirements. Multiple vendors were just introducing products that would support multi-protocol networks. To implement our networking solution all that we needed was a new version of Novell software on the file server and some software on the PC. The PC software allowed us the use the X-windows interface through Microsoft Windows. Thus PC users would have a Microsoft Windows environment with perhaps a word processor open in one window and SAS on the UNIX processor open in another window.

Benchmark Development and Results

Since our main thrust in buying a new system was improved response time for large datasets, we needed a way of measuring the speed of alternative solutions. The big variable in our conceptual solution was which UNIX processor we should get. We selected a benchmark set of four different SAS analysis steps: a sort, a generalized linear model analysis, a univariate analysis, and an analysis using PROC FREQ. This mixture of descriptive and analytic procedures approximated our usual workload. To establish a baseline, we ran these procedures on our existing system against five different-sized subsets of one of our large datafiles. The file sizes used were 1,000 10,000, 20,000, 100,000 and 200,000 records, with 9 variables per record.

We tried to select file sizes and procedures reflecting the type of work expected. We also wanted to see how the systems would respond to a range of conditions - would there be a breaking point? However, we had to limit the size of the benchmark so that the test files were portable to another facility and the test was able to be run in a reasonable amount of time (so as not to disrupt the test facility). We would have liked to have started out with a 1,000,000 record database - but we didn't have an easy way of transporting such a file. As it was, our initial benchmark set of four procedures and five file sizes took a total of 3 hours to execute on our existing 386 system (see figure 3 for times for specific analyses and file sizes.)

We then took our benchmark datasets and analysis commands and visited one hardware and one software vendor (who had several different brands of hardware available for demonstrating his software). Both vendors were extremely cooperative in supporting us in running the tests. The response times we measured on the new systems truly amazed us. Figure 4 shows data for one particular processor. Note that the longest analysis step has shrunk from about 62 minutes to about 1 minute. The total process took only 4.6 minutes on this processor (versus the

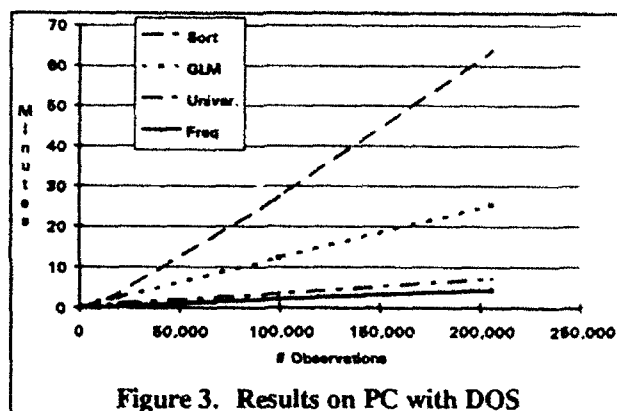


Figure 3. Results on PC with DOS

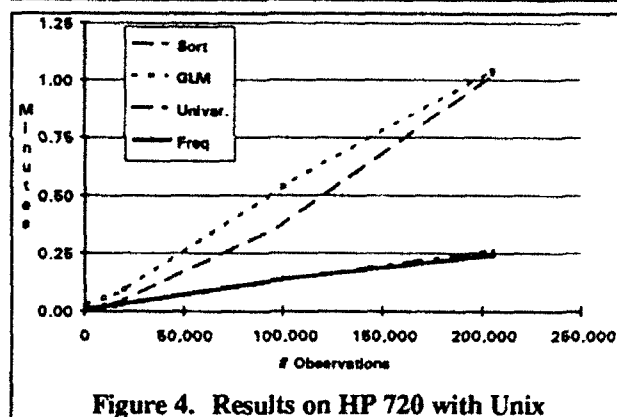
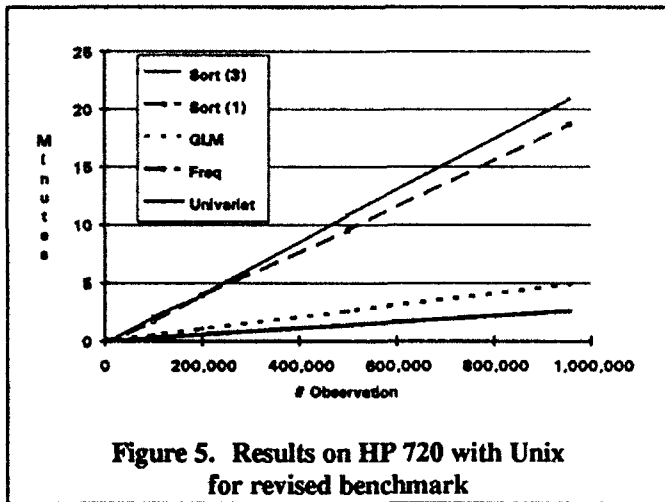


Figure 4. Results on HP 720 with Unix

3 hours on the baseline PC). This was not just an order-of-magnitude improvement, but was approaching two orders of magnitude!

The improvements we measured on our first set of tests caused us to go back and generate a more complex benchmark. The revised benchmark had slightly different procedures (e.g., adding a multi-key sort) and two additional file sizes (500,000 and 955,000 records). To move this amount of data to a test processor, we just moved our whole PC with its 300 MB disk to the vendor facility, connected it to their network, and then copied the data over to the test machine.

Figure 5 shows the results of this revised benchmark on the same processor tested above. Note that the longest analysis steps took about 20 minutes to process approximately 1 million records. Note also the essentially linear trend in response time as a function of file size. This linear trend is unexpected with procedures such as sorting. SAS documentation indicates sorting should require greatly increasing time as file size increases. No explanation for our measured behavior is immediately obvious.



A Few Last Notes on Benchmarking

- We measured performance on two different UNIX/RISC vendors' hardware, not on all possible candidates. One reason was that certain vendors did not have cost-effective hardware in the summer of 1991. There is one standard benchmark set that all UNIX/RISC vendors use to measure their performance - the SPECmark - and SPECmark ratings are a standard feature of advertising brochures. The SPECmark rating of the processor reported on here was 55. Our measurements on several processors indicated that the SPECmark ratings reasonably established relative performance. Using the SPECmark ratings we thus judged it wasn't worth considering the remaining candidates.

A second factor limiting testing was the lack of cooperation from some other vendors, particularly third party re-sellers. The lack of support and professionalism on the part of one of these made us appreciate even more the testing support we got.

- We never were able to run tests duplicating the environment in which we expected to run. All our tests were run completely within the UNIX/RISC processor. That is, we never were able to access SAS using X-Windows from a PC across a TCP/IP network. Our candidate solution required the products of multiple manufacturers and would thus be most likely in the province of a third-party re-seller. In our limited experience, such folks did not have any demo facilities and were not interested in working with us to do the testing. We were lucky that the hardware and software components we purchased did indeed work reasonably well together.

- The UNIX boxes we tested were multi-user/single processor systems. We did run a test simulating two

simultaneous users on the processors. The same script was run simultaneously against the same files. Not unexpectedly, the time for each to complete was about twice as long as for that of one executing alone.

Costs

The cost discussion here is limited not because cost is unimportant, but because it seems to be always changing. Our cost analysis was done in the summer of 1991 using standard vendor prices available through existing government contracts. All components of the system (e.g., processor and disks) were priced according to the vendor's published/quoted prices. Since our study, some vendors have gone through two significant price cuts and others have released new products.

We found a fairly wide range of costs across vendors, with vendors who had a recently released product being significantly lower in costs than those with products one or two years old. We also noted some vendors might be extremely uncompetitive in one component of the system (e.g., disks) and yet reasonably competitive on the remaining. In such cases, the buyer has the option of seeking a third-party supplier for his disks, but we did not consider this in our analysis.

With all those caveats, today (May, '92) a UNIX/RISC processor rated at 60 SPECmarks with 32 MB of RAM and 4 GB of disk lists commercially at about \$45,000 to \$50,000. The cost of a gigabyte of disk is about \$4,500 and 10 megabytes of RAM is about \$2,500. Smaller UNIX/RISC systems are available in the \$5,000 to \$15,000 range. Discounts of about 25% off these prices are reasonable for large institutions. The bottom line was that we were able to meet our needs within our budget.

Conclusions

Our small group of data analysts found that UNIX/RISC processors were a key part of a cost-effective computational environment for analyzing large data sets. We were happy to find that this product integrated well with our existing PC network. Groups seeking to improve their computational environment should carefully assess their requirements and then systematically compare them to what is on the market. Establishing a new environment requires considering a wide range of issues including not only the statistical functions needed, but also the user interface, the anticipated patterns of use, system performance and capacity, networking mechanisms, and how well statistical and non-statistical tools co-exist and co-operate.

REUSE-ORIENTED APPROACH IN DEVELOPING STATISTICS SOFTWARE

Zhiyi Lou and Antonio Ciampi

School of Computer Science, McGill University, Montreal, Canada

Abstract

The existing gap between demand and ability to produce statistical software cost-effectively calls for a reuse-based software development approach. The approach proposed in this paper, based on object-oriented design and programming, maximizes all levels of reuse and generates reuse-enabling software products. Three basic concepts concerning reuse are discussed: (1) domain-oriented software life cycle, (2) reuse-oriented software development process model, and (3) experience factory. A reuse-oriented development process approach derived from these concepts is outlined, and its implementation is demonstrated in the case of the development of a statistical software system, RECPAM, for constructing generalized regression trees from data.

1. Introduction

Owing to the increasingly important role played by computing in both theoretical and applied Statistics, the gap between demand for high-quality software and the ability to produce it cost-effectively and in a reasonably short time, is often decried. Statistics-specific languages such as S, are becoming increasingly popular, as are software packages with built-in programming capabilities. These tools, however, do not meet the need for developing certain types of specialized programs requiring a somewhat "open ended" design, i.e. the possibility of adding new, complex modules and/or modifying some portion of the system. What is needed, is rather an approach to programming that allows for reuse in a simple, straightforward way.

The concept of software reuse has appealed to programmers since the creation of the first stored-program computer [1]. It is behind almost the every software development. Since McIlroy 1969 proposal [2] of establishing a software components catalog, from which software parts could be assembled, much as is done with mechanical and electronic components in engineering, more and more people have started developing software with reuse particularly in mind. With the success of the Japanese software industry, a number of industrial organizations began to focus on reusability. As a result, various reuse methodologies have been proposed, and it is predicted that software reuse will probably be crucial to the evolution of the software industry towards higher levels of maturity [3].

These developments in software engineering are very

relevant to statistical applications, since in many cases the following conditions, believed to be decisive factors to foster successful software reuse [4], are satisfied: (i) The domain is relatively narrow (it contains a fairly small number of data types); (ii) The domain is well understood (it has evolved over hundreds of years); (iii) the underlying technology is quite static (it has reached a high level of maturity). It should therefore be clear that many computer intensive statistical methodologies provide a fertile ground for comprehensive reuse-based approaches. Nevertheless, reuse is only exploited to a limited extent in most statistical software systems. At any rate, the situation is far from ideal. Ideally, a reusable software system should be more like hardware. There should be catalog of software modules, which include all kinds of software-related experience, as there are catalog of VLSI devices. When building a new system, one should be able to order components from catalogues and to assemble them, rather than reinventing the wheel every time.

The goal of this paper is to provide an introduction to software reuse concepts for Statistics. After presenting the motivations for a reuse-oriented approach, we discuss three technical concepts for supporting reuse, and develop from them a new reuse-oriented software development process. As an application of this process, we illustrate the implementation of a statistical system for tree-growing, RECPAM.

2. Reuse-oriented approach: Motivation

Software reuse is the reapplication of knowledge about one system to another similar one in order to reduce the effort of development and maintenance. This reused knowledge includes artifacts such as domain knowledge, development experience, design decisions, architectures, requirements, design, code, documentation and so forth. Without a systematic approach, however, several fundamental technical problems limit software reuse in practice. We will focus on some of them.

i) Multiorganizational problem

Software systems are not often initially designed for future reuse, since the emphasis is on meeting specific project requirements. Optimal pursuit of a specific project's objective is rarely compatible with decomposition into reusable modules and packaging of

relevant programming experience in a generalizable form. To develop a reuse-enabling system, reusability must be engineered from the start and the objective of the development must have a multiplicity of objectives, a concern known as *multiorganizational problem* [4].

ii) Leverage problem

Leverage refers to the degree to which a reused module reduces the effort needed to produce new software. Leverage varies with physical size of the module, level of module abstraction and level of machine processability of the module [5]. As abstraction and size of the software module increases, enhancing reusability, processability usually decreases.

There are three levels of abstraction in software reuse: code reuse, design reuse and knowledge reuse. Current reusable modules, reusable building blocks and reusable patterns, concern code and, to a limited extent, design reuse. Their corresponding size is small. Ideally, reuse should support all levels of abstraction in a richly machine-processible form.

iii) Operational problem

Software reuse is not a specific technique, algorithm, heuristic or set of guidelines. It is many different mixtures of technologies, process modules and cultures. This demands a radical departure from the operational style prevalent in current programming. Much of the current work in reuse focuses on a particular phase without addressing the transition and traceability. Most of the current systems are constrained to apply a few specific reuse techniques or mechanisms without synthesizing them into a consistent approach.

The above discussion emphasizes the need for the systematic approach known as reuse-oriented approach. Recent enabling methodologies for this approach are object-oriented design and programming. Object-oriented design facilitates the integration of analysis, design and programming within a single framework using common concepts and (often) notation. It provides a high-level primitive notion of modularity for directly modelling application [6]. Object-oriented programming supports many reuse mechanisms: object class, instantiation, inheritance, polymorphism, overloading and generic classes, which promote larger and more abstract reusable components [7].

3. Reuse-oriented approach: Methodology

The reuse-oriented approach presented in this paper is based on three technical concepts which arise as mixtures of technologies, process models and cultures.

3.1 Domain-oriented software life cycle

The single project life cycle is limited to scope and

abstraction level of reuse modules. In contrast, domain-oriented software life cycle extends to a "domain", i.e. a designated collection of existing applications and anticipated opportunities for future applications with common functionality in one or more areas. "A domain life cycle model formalizes typical patterns (of reusable experience) in the development of a related series of projects and the persistence of information (on reusable components) from one application to the next." [8]. The multi-project view is an essential foundation for a general model of reusability, which supports to capture and reuse domain-specific knowledge and implementation knowledge across applications. According to the domain-oriented software life cycle model, reuse-oriented software development assumes that, given an application domain, there is a general system experience base. If a new project in the domain is needed, it can be assembled by extracting modules from the experience base and generating some project specific modules, instead of creating all modules from scratch. Thus, software reuse fosters informal sharing of software related experience among people working on similar projects within an application domain.

3.2 Reuse-oriented software development process model

An evolutionary model is needed that enables organizations to learn from each individual project, and incrementally to improve their ability to contribute to the whole domain. This implies a significant cultural change among software developers, from a project-oriented process model to reuse-oriented process model. The traditional project-oriented development process attempts to accomplish more with less resources, i.e. deliver the required systems faster, reduce turn-around time in maintenance, increase performance, reliability, and security of systems. The reuse-oriented development process, on the other hand, aims at improving the effectiveness of the process, reduces the amount of work, and reuses life cycle products [9]. Based on the domain-oriented software life cycle assumption, a model used for reuse-oriented approach splits the traditional life cycle model into two coordinated organizations: an experience-packaging organization, and a project-generating organization. The experience-packaging organization's primary concerns are the recognition of potentially reusable experience appropriate in an application domain and packaging them in a readily available way. The project-generating organization develops products taking advantage of all forms of packaged experience from prior and current development, while offering its own experiences to be packaged for other projects.

3.3 Experience factory

The experience factory is a key ingredient to the reuse-

oriented approach. It is a rich and well-organized framework. The experience factory functions as a target which can learn experience from domain analysis and individual development for the experience-packaging organization. For the project-integrating organization, it functions as a source of experience for constructing new projects. To make software-related experience reusable, it must satisfy four characteristics: (1) generality, (2) definiteness, (3) transferability, and (4) retrievability [10]. Packaging reusable experience involves three phases:

i) Abstraction: Abstraction characterizes a class of entities by their common attributes and ignores, for the time being, their differences. Every abstraction determines a set of associated reusable attributes. Conversely every set of reusable attributes determines an abstraction - the class of entities that possesses these attributes [10]. Many different abstraction mechanisms have been proposed as the basis for a software module industry, each representing different building modules from which programs can be constructed and each resulting in different paradigms for programming. We adopt three abstraction: (1) function abstraction, (2) data abstraction and (3) process abstraction.

ii) Classification: Classification is grouping similar things together. All members of a group produced by classification share at least one characteristic that members of other classes do not. The result is a framework or structure of relationships [11]. Software development is an iterative refinement process in which requirements specified in an application domain are gradually transformed into programs to be executed on a target computer. Transformation in a single step is impossible in a larger system. In general, the transformation is divided into five steps, we classify the software-related experience into corresponding five levels: (1) environmental-level information, (2) knowledge of application domain and development, (3) functional architectures (external specifications of system functions and data), (4) logical structures (internal designs of processes and data structures), and (5) code fragments (executable subroutines), so that experience factory is able to support five levels of reuse: application model reuse, knowledge reuse, specification reuse, design reuse and code reuse.

iii) Representation: The ideal representation must allow the specification and storage of such partial architectures, and it must allow incremental completion of the details over time. Object-oriented design and programming is the best candidate, because it supports the following: the ability to represent knowledge about implementation structures in factored form; the ability to create partial specifications of design information that can be

incrementally extended; the ability to allow flexible couplings between instances of designs and the various interpretations they can have; and the ability to express controlled degrees of abstraction and precision.

3.4 Reuse-oriented approach: paradigm

The development scheme is characterized as two interrelated organizations, the experience-packaging organization and the project-generating organization, which incorporate the three technical concepts. As depicted in Fig. 1, the two organizations both have three basic activities. The experience-packaging organization includes domain analysis, experience abstraction, and experience representation.

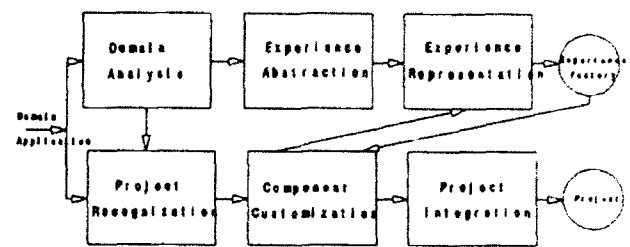


Fig.1 Reuse-Oriented Approach Paradigm

i) Domain Analysis

Domain analysis can be conceived of as a knowledge intensive activity occurring prior to the experience abstraction phase to identify all kinds of experience within a domain as reuse candidates, and prior to the project recognition phase of project-generating organization to instantiate a domain model to a specific application. As transcending specific applications, it is at a higher level of abstraction than system analysis. In domain analysis, common characteristics from similar projects are generalized objects, operations and processes common to all projects within the same application domain are identified, and a model and interface are defined to describe their relationships. The next step is to define a domain specific language and use it to describe our domain model.

ii) Experience Abstraction

The objective of abstracting identified experience is to make a candidate reuse object useful in a large set of potential target applications. Function, data and process abstraction determine different paradigms for programming associated with different partitioning of a computation into reusable and varying parts. Function abstraction emphasizes the reusability of functions of varying data; data abstraction emphasizes the reusability of data objects for various operations that may be applied to them, while process abstraction emphasizes data

objects for various operations that have an independently executing thread of control that determines the order in which operations become available for execution.

iii) Experience Representation

The objective of recording experience is to create a repository of well specified and organized experiences by encoding them in more precise, better understood ways. It provides a reuse-enabling product, experience factory.

The project-generating includes project recognition, component customization, and project integration.

i) Project Recognition

Based on the domain model from domain analysis, the objects, operations and processes matching the domain model are recognized for retrieving the reuse candidates. At the same time, the requirements, objects, operations and processes specific to a concrete project are recognized for considering the new experience to be packaged. It is an extension of domain analysis.

ii) Component Customization

Component customization is intended to bridge the gap between retrieved reuse candidates and given reuse requirements. It is the lifeblood of reusability. It includes two branches: identifying and customizing the reuse candidates, and adding new experience or refining the existing experience.

iii) Project Integration

In order to deliver the system, project integration assembles all customized experience using a few composition mechanisms, such as, pipe, message-passing, inheritance and so on. Then it continues as usual with product quality control and release.

4. Case study: The development of a statistical application-RECPAM

RECPAM is a tree-growing methodology proposed by Ciampi *et al.* [12, 13]. A tree is constructed from a data set: it represents the knowledge contained in some variables, called predictors, about a parameter, called criterion. This parameter is estimated from other variables, called criterion variables. RECPAM constructs the tree in three steps: (i) RECURSIVE partition, (ii) Pruning and (iii) AMalgamation (hence the acronym). The resulting classes are described by conjunctive and disjunctive statements involving the predictors, and represents subpopulations for which a distinct estimate of the criterion is called for. The classes are represented as a regression model for the criterion, with class indicator functions as regressors. It follows that for every regression model, one can build a RECPAM tree. Indeed, a corresponding tree approach has been developed for regression within the Generalized Linear Model (GLIM),

the Cox model for the analysis of censored survival data with covariates and, more recently, for multivariate normal regression.

Since RECPAM is a general methodology with open ended applicability, it is of interest to develop a program that the user may wish to modify or extend to include other regression models. We are developing a new version of RECPAM by the reuse-oriented software development approach outlined above. Considering this approach, we choose BORLAND C++ as programming language, which includes most techniques and mechanisms supported by object-oriented programming, and use ObjectWindow for supporting user interface design and M++ for supporting numerical computations. These two object libraries are designed as the base of RECPAM experience factory.

The development starts from *domain analysis*. The principle of the RECPAM is seen as recursively constructing a partition with maximal local information content to build a large tree, followed by eliminating in turn "negligible" information with minimum global information loss respectively to get the honest tree and the RECPAM partition. The measure of information in three steps is based on estimating the model parameters and computing model regression. From the viewpoint of statistics, different statistical models share the same RECPAM manipulations and only change the information measure which comes from the regression. It is logically and ideally suited to separate the regression from recursive partitioning, pruning and amalgamation. Also, we can unify the data formats from different statistical models to two groups: predictors and criterion. The RECPAM implementation prototype consists of five modules, as shown Figure 2. The five modules are ranked into three domains: regression, RECPAM, and data handling. Each domain is an independent system which can be directly executed or reused by other systems as a subsystem. For example, the regression module is not specific for the RECPAM approach. According to the three domains, the experience factory is organized into three corresponding packages. For each domain, we identify common behaviours across different models and specify the interconnection between two domains. The RECPAM domain is divided into three steps: tree building, tree pruning and amalgamation.

In *experience abstraction*, we try to describe the results of domain analysis by means of data abstraction, function abstraction and process abstraction. For example, the RECPAM domain deals with three types of data objects: data matrices, tree structures and partition structures. Each data object has operations associated with it. For example, some operations associated with data matrices

are maximizing the information content, splitting the data matrix into two partitions, surrogating the missing data.

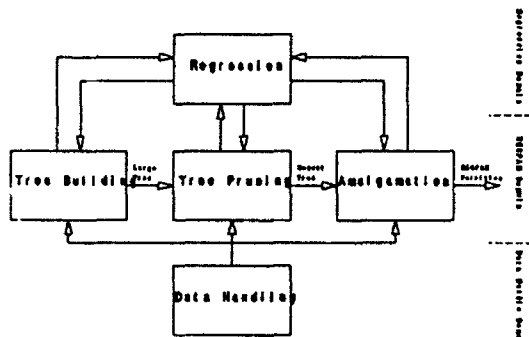


Fig.2 RECPAM Implementation System

In *experience representation*, three abstractions are mapped into programming codes, specifications and documents. Then they are organized into a framework by describing constraints such as precedence, sequencing and synchronization on the possible sequence of actions during the execution of RECPAM.

If a particular statistical model is needed, we recognize the similarities and differences. The similarities, such as, RECPAM approach, Newton-Raphson method, preparing data can be reused by retrieving and customizing experience from the experience factory. The differences, such as, each model's regression, criterion variables declaration and stopping rules, need to be created. As new experience, these new parts are packaged for future applications. Finally, we assemble and test them to deliver the new model in the RECPAM family. This approach significantly decreases duplication of work. It promotes productivity, reliability, consistency, manageability and standardization.

5. Conclusions

We have introduced a reuse-oriented approach, synthesizing three reuse technical concepts: domain-oriented software life cycle, multiorganization development process model and experience factory into conventional development process, to enhance the reuse level in developing statistics software. It provides a systematic approach to maximize reusing all software-related experience.

We have demonstrated the development of RECPAM, as an example of the reuse-oriented approach. The experience confirms the following points:

- Software reuse changes the software development process.

- Technology is important and infrastructure support is essential.

- Narrow well understood domains are more effective.

- Domain analysis is at the heart of the reuse-oriented approach.

References

- [1] Tracz, W., "Software Reuse Myths", ACM SIGSOFT Software Engineering Notes, Vol. 13, No. 1, Jan. 1988, pp 17-21.
- [2] McIlroy M., "Mass Produced Software Components", Proc. NATO Conf. on Software Eng., Petrocelli/Charter, New York, 1969, pp. 88-98.
- [3] Caldiera, G. and Basili, V.R., "Identifying and Qualifying Reusable Software Components", IEEE Computer, Feb. 1991, pp. 61-69.
- [4] Biggerstaff T.J. and Richter C., "Reusability Framework, Assessment, and Directions", IEEE Software, Vol.4, No.2, Mar. 1987, pp. 6-16.
- [5] Biggerstaff, T.J. and Perlis, A.J., "Forward: Special Issues on Software Reusability", IEEE Transactions of Software Engineering, Vol. 10, No. 5, 1984, pp. 474-476.
- [6] Hopkins, T.P., "Object-Oriented Systems", Software Engineering Journal, Vol. 7, No. 2, Mar. 1992, pp. 82-83.
- [7] Wegner, P., "Concepts and Paradigms of Object-Oriented Programming", OOPS Messenger, Vol. 2, No. 3, Jul. 1991, pp. 8-89.
- [8] Simos M.A., "The Domain-Oriented Software Life Cycle: Towards an Extended Process Model for Reusability", Proceedings of the Workshop on software Reusability and Maintainability, Oct. 1987.
- [9] Basili, V.R. and Caldiera, G., "A Reference Architecture for the Component Factory", Computer Science technical Report, Mar. 1991.
- [10] Matsumoto, Y., "Some Experience in Promoting Reusable Software: Presentation in Higher Abstract Levels", IEEE Transactions on Software Engineering, Vol. SE-10, No. 5, Sep. 1984, pp. 502-512.
- [11] Prieto-Diaz, R. and Freeman, P., "Classifying Software for Reusability", IEEE Software, Vol. 4, No. 1, Jan. 1987, pp. 6-16.
- [12] Ciampi, A., "Generalized Regression Trees", Computational Statistics & Data Analysis, Vol. 12, 1991, pp. 57-78.
- [13] Ciampi, A., Lou, Z., Lin, Q., and Negassa, A., "Recursive Partition and Amalgamation with the Exponential Family: Theory and Applications", Applied Stochastic Models and Data Analysis, Vol.7, 1991, pp. 121-137.

Systematic Editing of Complex Questionnaire Skip Patterns.

Robert F. Teitel
 Abt Associates Inc
 4800 Montgomery Lane
 Bethesda, MD 20814

I. Abstract.

Large surveys contain complex skip patterns: a reply of {no} to "ever worked?", for example, would skip over the entire job data section; similarly, each of the responses to "how often paid?", ({by hour}, {by day}, {by week}, etc.), would skip to appropriate questions on earnings. These skip patterns can overlap and be nested in complex ways. Computer Assisted Telephone Interviewing (CATI) systems contain the necessary logic to implement the skip patterns. Non-CATI data from surveys with large pre-test and/or field follow-up require editing of skip pattern violations.

This presentation will describe the relevant theory and the implementation of a systematic skip pattern editing process and its application to a survey with 25,000 observations and over 1,000 variables.

II. Introduction.

Skip patterns are used in survey instruments essentially for three purposes: to skip questions which do not apply to the respondent, to skip questions whose answers can be deduced from answers to prior questions (though some redundancy is often used for internal consistency checking), and to elicit answers to alternate questions for those questions the respondent refused or did not know the answer.

Large-scale surveys are now usually administered using Computer Assisted Telephone Interviewing (CATI). The survey instrument is encoded in the processing language of the CATI system, complete with skip pattern control. Alternate paths through the questionnaire are taken as a function of the responses the CATI operator enters.

For an employment and training program evaluation study, a series of large survey instruments were administered to economically disadvantaged Americans. A substantial portion of this data collection effort could not be completed over the telephone and had to be administered in person by survey field staff using paper booklets. These booklets were "maximally" key-entered: all responses in the booklet were entered regardless of context. The data derived from this process will contain skip pattern violations and will require data "cleaning" before analysis commences.

III. An Example of the Problem.

Parts of a survey instrument constructed to illustrate typical skip patterns is shown in Figure 1.

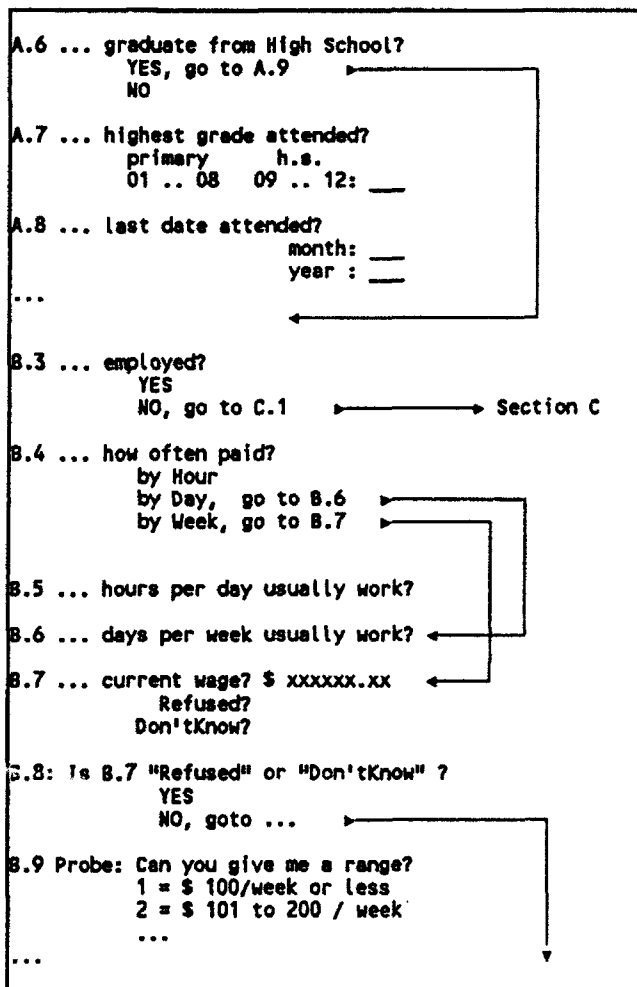


Figure 1: Skip patterns in a survey instrument.

A skip pattern violation exists, for example, in a data record where A.6 is {YES} and both A.7 and A.8 contain valid values -- a High School graduate who should have skipped A.7 and A.8. Similarly, a skip pattern failure exists if B.4 is {by hour} and B.5 is {blank} -- an hourly employee should answer the hours per day question.

III. Traditional Methods.

Traditional methods of cleaning skip patterns consist of studying multi-dimensional tabulations and/or customized computer programs to determine the extent and context of skip pattern errors. Research personnel then determine the appropriate corrections or imputations for the variables involved in the patterns, often on a case-by-case basis.

The length and complexity of our follow-up survey instruments (60 pages, 1,200 variables) and the anticipated 25,000 respondents suggested that alternate, more systematic and automatic processing approaches should be investigated.

An informal literature review and discussions with personal contacts found only one relevant reference: [Fagan and Greenberg, 1988].

IV. Systematic Approach to Editing Skip Patterns.

Skip patterns in survey instruments can clearly be modeled with *acyclic directed graphs*: Questions are represented by *nodes*; values are represented by *edges* or *arcs*. The graph model is *acyclic* because survey instruments do not have the potential of unending loops (although the instrument may repeat a series of questions "for each job"), and the model is *directed* because each arc has a distinct, unique, starting node and a distinct, unique, terminal node. A survey instrument usually begins with a single starting question (or node) and, if not terminated with a single terminating question ("Thank you"), can trivially be made to do so.

We will use the simple graph in Figure 2 to demonstrate our systematic approach to skip pattern editing.

The sample graph can be described with an arc *adjacency* matrix wherein each cell indicates whether the row arc connects to the column arc as illustrated in Figure 3.

Fagan and Greenberg proved that, given an adjacency matrix, M , the matrix obtained by $M \uparrow 2$ gives "2-step" connectivity, the matrix $(M \uparrow 2) \uparrow 2$ gives "4-step" connectivity, etc., with the matrix $M \uparrow \text{ceil}(\log, N)$ gives "N-step" connectivity. "N-step" connectivity, or all possible paths from the initial to the terminal node is identical to *transitive closure*; a simple, yet fast, algorithm for transitive closure is easily derived, and is credited to [Warshall 1962] by [Sedgwick 1990].

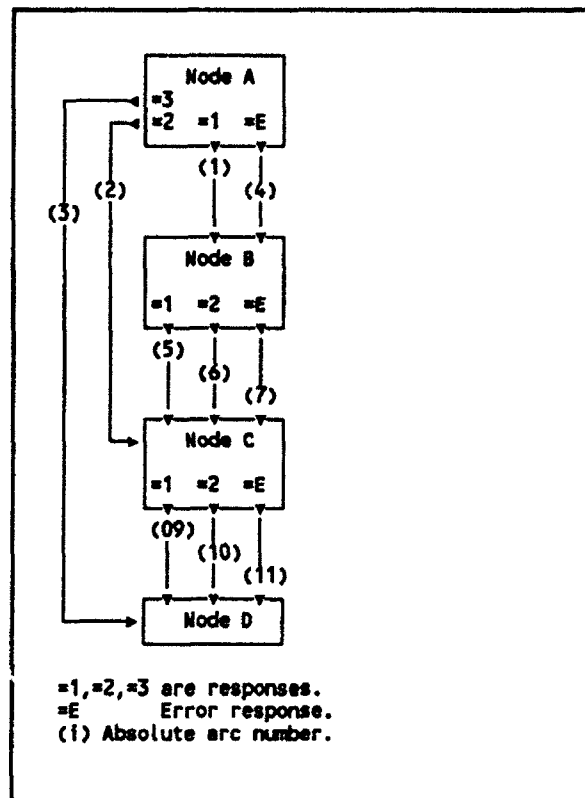


Figure 2: Skip pattern graph model example.

arc	1	2	3	4	5	6	7	8	9	0	*
A=1	1				1	1	1				
A=2		1						1	1	1	
A=3			1								1
A=E				1	1	1	1				
B=1		5			1			1	1	1	
B=2		6				1		1	1	1	
B=E		7					1	1	1	1	
C=1		8						1			1
C=2		9							1		1
C=E		0								1	1
D	*										1

Figure 3: Arc adjacency matrix.

Fagan and Greenberg, in their paper and in their computer programs which they kindly shared with me, represented {blank} implicitly: an absence of any arc for a node represented a blank value. I found it more convenient to encode {blank} as an explicit arc from the nodes permitting {blank}.

The transitive closure matrix, T-matrix, for the skip pattern graph model example in Figure 2 extended with explicit Blank arcs, is as follows:

arc	1	2	3	4	5	6	7	8	9	10	11	12	*
A=1	1				1	1	1		1	1	1		1
2	2	1					1		1	1	1		1
3	3		1				1					1	1
E	4			1	1	1	1		1	1	1		1
B=1	5	1		1	1				1	1	1		1
2	6	1		1	1				1	1	1		1
E	7	1		1			1		1	1	1		1
8	8	1	1				1		1	1	1	1	1
C=1	9	1	1	1	1	1	1	1	1				1
2	10	1	1	1	1	1	1	1	1	1			1
E	11	1	1	1	1	1	1	1			1		1
8	12		1				1					1	1
D	*	1	1	1	1	1	1	1	1	1	1	1	1

Figure 4: Transitive closure matrix.

Every data record represents a path from the initial node to the terminal node. Whether that path is a valid path can be determined from the T-matrix. For example, a record with values (A = {2}; B = {blank}; C = {1}; D) is encoded as the path (arcs (2), (8), (9)). The data path connections (2) → (8) and (8) → (9) exist in the T-matrix; Hence the record "passes".

Similarly, a record with (A = {1}; B = {blank}; C = {2}; D) is encoded as (arcs (1), (8), (10)). The sample skip pattern T-matrix from Figure 4 is shown in Figure 5 overlaid with the path (arcs (1), (8), (10)) from this data record. The connection (1) → (8) in the data does not exist in the T-matrix and, hence, a skip pattern violation exists.

V. Imputation Heuristics.

The use of the T-matrix provides a mathematically sound method for determining the correctness of the Node-Arc path representation of any data record. The second phase of the systematic editing process, imputation, is invoked once a record is determined to contain at least one skip pattern violation.

arc	(1)	2	3	4	5	6	7	(8)	9	(10)	11	12	*
A=1	(1)				1	1	1	()	1	(1)	1		1
2	2	1						1	1	1	1		1
3	3		1					1				1	1
E	4			1	1	1	1		1	1	1		1
B=1	5	1		1	1				1	1	1		1
2	6	1		1	1				1	1	1		1
E	7	1		1			1		1	1	1		1
B	(8)	()	1				(1)		1	(1)	1	1	1
C=1	9	1	1	1	1	1	1	1	1				1
2	(10)	(1)	1	1	1	1	1	(1)		(1)			1
E	11	1	1	1	1	1	1	1			1		1
B	12		1				1					1	1
D	*	1	1	1	1	1	1	1	1	1	1	1	1

Figure 5: Sample T-matrix with path overlay.

Fagan and Greenberg note that their skip pattern software "is not meant to be a comprehensive edit and imputation package." The software removes a minimal set of responses from records with skip pattern violations, distinguishes non-applicable from missing items, and performs response imputations for missing--but needed--items. They suggest that "one can frequently discover deterministic imputations for selected missing items."

Whereas Fagan and Greenberg base most of their imputation strategy on *arc* conflict counts, our process determines a conflict participation count for each *node*. Based on then conflict count node structure we apply one of a set of three imputation schemes to the record.

The imputation schemes were developed heuristically to satisfy three criteria: 1) to do the least possible damage to the data, 2) to be extremely conservative in imputing valid value arcs to nodes (answers to questions), and 3) to match the imputations which would have been made manually.

The three schemes are as follows:

- a. A unary conflict count mode.
When there is a single node with the highest conflict count, we determine the conflict count for each of its arcs. If there exists a single arc with minimal conflict, we impute that arc to the node. This is the only occasion where we will impute an actual data value. If there are two or more minimal conflict count arcs we impute {error}.

- b. A conflict count mode with three or more nodes. We set each node to {error}.
- c. A bi-modal conflict count. When there are exactly two nodes with the highest conflict count, the imputation scheme is based on the range of possible arc states and on the current arc state of each of the two nodes. The possible arc states are {value / no-blank, value / blank-ok, blank, error}, where value / no-blank indicates that a node has a non-blank valid value and that blank is not a permitted state, and value / blank-ok indicates that a node has a non-blank valid value and that blank is a permitted state.

The imputation scheme is shown in Figure 6 as state transition matrix.

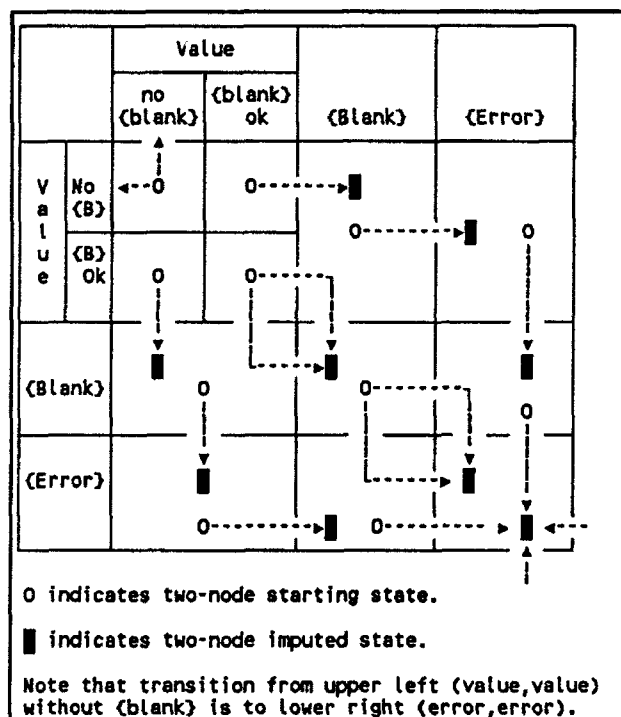


Figure 6: Bi-variate state transition imputation scheme.

The transitions between the bi-variate starting and imputed states do not permit imputing (new) valid values to either of the nodes involved. One could take a much more permissive view of data imputation and have transitions "up" and "left" instead of our "down" and "right"; we have not tested such a scheme.

VI. Implementation.

After implementing a small prototype skip editor to test the basic approach, the full skip pattern editor was implemented (in FORTRAN 77) "on top of" an existing univariate editor: the skip editor is then assured of valid data arcs including {blank} and {error} arcs where appropriate. The skip pattern structure is derived from an existing codebook management program.

The full production version was much more complex and took much longer to develop than anticipated. Nevertheless, it was successfully used on segmented parts of the follow-up surveys. Editing was done on an (IBM) mainframe and on various PCs.

The size of the T-matrix, regardless of the degree of "bit-packing", is a function of the square the number of unique arcs (not nodes) in the graph of the skip patterns in a survey instrument. Other implementation data structures are linear functions of the number of arcs and of the number of nodes.

The full production version can edit data segments with approximately 100 nodes (variables) and 500 arcs (values).

Several modifications to the program are being studied, especially space-conservation changes.

VIII. Results.

Systematic editing of complex skip patterns in survey data is possible, and it has been implemented for use in a production environment. The detection of skip pattern violations is based on the properties of the transitive closure matrix of the graph model of the skip patterns and a set of heuristically derived schemes for skip pattern imputation has been developed. The results of the editing and imputation process are identical or equivalent to those produced manually.

IX. References.

Fagan, Jim, and Greenberg, Brian, *Using Graph Theory to Analyze Skip Patterns in Questionnaires*, SRD Research Report Number Census/SRL/RR-88/06, Bureau of the Census, Washington, DC, 1988.

Sedgewich, Robert, *Algorithms in C*, Addison-Wesley, Reading, MA, 1990.

A Program for Identifying Duplicated Code

Brenda S. Baker
AT&T Bell Laboratories
Murray Hill, New Jersey 07974

ABSTRACT

This paper describes a program called *dup* that finds occurrences of duplicated or related code in large software systems. The motivation is that duplication may be introduced into a large system as modifications are made to add new features or to fix bugs; rather than rewrite working sections of code, programmers may copy and modify sections of code. Over time, proliferation of copies can make the code more complex and more difficult to maintain. *Dup* searches such code for all pairs of duplicated sections. The user may choose to search either for identical sections of code, or for sections that match except for substitution of one set of variable names and constants for another as if they were corresponding procedure parameters. Applications of *dup* could include visualization of the structural complexity of the whole system, identifying unusually complex files, identifying sections of code that should be replaced by procedures, and debugging.

Introduction.

This paper describes a new software tool, a program called *dup*, that finds occurrences of duplicated or related code in large software systems to aid in software maintenance and debugging. The program generates descriptions of the related code sections and statistics about the extent of duplication found. For visualizing the output, the output can be plotted in scatter plots and profiles can be produced to show how many times each line occurs in matching sections of code.

This work is part of the emerging field of *software visualization* [E], whose goal is to display characteristics of large software systems visually, as an aid in dealing with the complexity arising in systems of hundreds of thousands or millions of lines of code created by hundreds or thousands of programmers. Other examples of software visualization are the graphical user interfaces *seesoft* [E] and *dotplot* [CH]. *Seesoft* interactively displays data such as the age or programmer for each line of code; *dotplot* allows interactive manipulation of scatter plots to compare sections of code.

Dup was motivated by the observation that duplication may be introduced into a large system as modifications are made to add new features or to fix bugs. Rather than rewrite working sections of code, programmers may copy and modify sections of code. It has long been known that copying sections of code may make the code larger, more complex, and more difficult to maintain. In particular, when a bug has been found in one copy, a bug fix may be made to the place where the bug was found, but not to the corresponding parts of other copies. Nevertheless, making a copy and modifying it may be much simpler than more major revisions and therefore less likely to introduce new bugs immediately, and hence copying code may seem preferable to changing a working section of code. This may especially be true when the programmer making the bug fixes is not the one who wrote the original code.

The premise underlying *dup* is that copying is most often accomplished by means of an editor. Therefore, the resulting copies will be largely the same line-for-line, or will be related in some systematic way such as a change of variables. White space and comments may be ignored since they do not affect the functionality of the code.

Given these assumptions, the approach taken in *dup* is line-based. Two lines of code are considered to be identical if they contain the same sequence of characters after removing comments and white space; the semantics of the program statements are not analyzed. Data structures are maintained with regard to lines rather than individual characters to reduce the space requirements.

The output of *dup* is a set of pairs of *longest matches* of sections of code. That is, two sections are a longest match if they match but the preceding lines do not match and the following lines do not match. To provide an example, we rephrase the definition of longest matches in terms of strings. Two identical substrings are a longest match if the preceding characters do not match and the following characters do not match. Thus, in the string *axyzbxyzc*, the *xyz*'s are a longest match, but the *xy*'s and the *yz*'s are not. Note that the longest match relation is not transitive. That is, if section A is a longest match for section B, and section B is a longest match for section C, it could be that section A is not a longest match for section C,

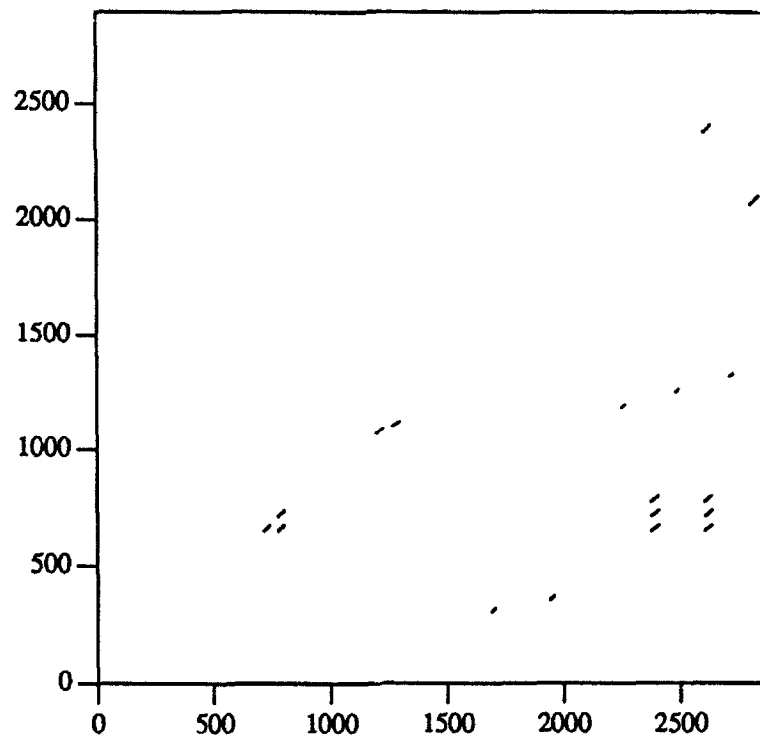


Figure 1. Exact matches for a C file.

because these sections are both contained in a longer match. In practice, when there are several related sections of code, the various pairs do differ in exactly how much matches. Therefore, the program reports the longest matches in pairs rather than looking for larger sets, and the scatter plots make evident that some of these matches overlap.

Since very short matches may not be interesting, the user may specify a minimum length match to report. Figure 1 shows a scatter plot produced by the program for a file of 2846 lines, or 1761 lines after pruning white space and comments, with a minimum match length of 15 lines. Only line segments below the main diagonal are plotted in this paper, because in plots of large amounts of code, most of the line segments are very close to the main diagonal, even though no line is matched with itself. Thus, each longest match is represented by exactly one roughly diagonal line in the plot; the lines are not strictly diagonal because the white space and comments have been ignored, while the line numbers are the original line numbers in the file. In this case, the program found 18 matches involving 419 lines, or 24% of the file.

Rather than looking for just exact matches, the program can look for *parameterized* matches, where the code sections match except for a one-to-one correspondence between candidates for parameters such as variables,

constants, macro names, and structure member names. (Keywords and operators are not candidates to be parameters.) For example, the following two code fragments taken (with some editing in order to fit) from the X Window [SG] source code are identical except for the correspondence between the variable names *pfi*/*pfh* and the pairs of structure member names *lbearing*/*left* and *rbearing*/*right*.

```
copy_number(&pmin, &pmax,
            pfi->min_bounds.lbearing,
            pfi->max_bounds.lbearing);
*pmin++ = *pmax++ = ',';
copy_number(&pmin, &pmax,
            pfi->min_bounds.rbearing,
            pfi->max_bounds.rbearing);
*pmin++ = *pmax++ = ',';

copy_number(&pmin, &pmax,
            pfh->min_bounds.left,
            pfh->max_bounds.left);
*pmin++ = *pmax++ = ',';
copy_number(&pmin, &pmax,
            pfh->min_bounds.right,
            pfh->max_bounds.right);
*pmin++ = *pmax++ = ',';
```

For parameterized matches, matching sections are like expansions of the same macro with different parameters, for

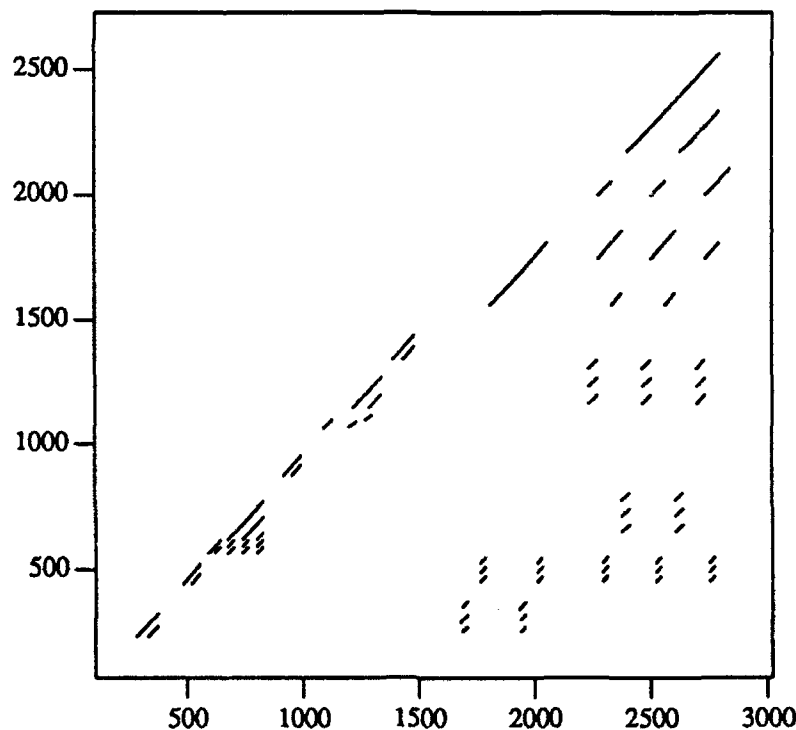


Fig. 2. Parameterized matches for the same file as Figure 1.

example, $f(p_1, \dots, p_n)$ and $f(q_1, \dots, q_n)$. Only pairs of parameters that are not identical need be reported. When run on the original X source containing the above code fragments, the output produced by dup is the following. (The line numbers given are the original line numbers; "pruned" lines are the ones remaining after stripping off white space and comments.)

```
xlsfonts.c:274-309,
fslsfonts.c:384-419,
34 pruned lines match, with parameters:
1: pfi, pfh
2: lbearing, left
3: rbearing, right
```

Commonly, code will have more parameterized matches than exact matches. For example, Figure 2 shows the parameterized matches for the same file as Figure 1. In this case, the program finds 87 longest parameterized matches of at least 15 lines, involving 85% of the file. The longest match found is 182 lines, compared to 37 lines for the exact matches.

The program makes an estimate of how much more succinctly the code could have been written, if alternative programming methods such as procedures had been used instead of copying. The estimate is based on the simple assumption that if the same line appears in k sufficiently

long matching sections of code, then $k-1$ of these occurrences could have been avoided. For the file of Figures 1 and 2, the program estimates that the code could have been shrunk by 14% based on exact matches, but 61% based on parameterized matches.

The program can also provide other aids such as a profile of the code showing how many copies of each line occurred in the matches found.

Other researchers have taken different approaches to finding common code. Programs aimed at detecting student plagiarism have typically used statistical comparisons of style characteristics such as the use of operators, use of special symbols, frequency of occurrences of references to variables, or the order in which procedures are referenced [H,Ja]. Johnson [Jo] has taken a parse-tree based approach to finding duplicated code. However, a line-based approach has two advantages compared to a tree-based approach. First, the line-based approach allows for use of data structures and efficient algorithms developed for string pattern matching, notably the suffix tree [McC]. Moreover, if the code contains macros, and some of these have, for example, unbalanced parentheses, it may be necessary to expand the macros to obtain a valid parse tree for the tree-based approach. However, the macros need not be

expanded for a line-based approach; thus, there is no need to rematch expansions of the same macros, and the resulting output is more easily related to the original code.

Church and Helfman [CH] have combined signal processing techniques with a graphical user interface **dotplot** enabling easy manipulation of scatter plots to aid in visually scanning for similarities between sections of code. By running **dup** and **dotplot** on the same input and overlaying the plots, it was found that some duplication may be found by means of both programs, while each program finds features that are not found by the other. **Dup** finds parameterized matches not noticeable in **dotplot** and eliminates the noise that appears in **dotplot** plots, while patterns prominent in **dotplot** plots may reflect characteristics such as repetitive control flow structure rather than duplication in the sense of **dup**.

Other related work has been done in the areas of file comparison, genome sequencing, and data compression. The UNIX **diff** program for file comparison [KP] and algorithms for genome sequencing [SK,LMW] have been based on finding a best match (*longest common subsequence* or *smallest edit distance*) between two sets of data, from start to end, whereas **dup** looks for pairs of shorter related sections that could appear in any order. Data compression algorithms such as [RPE,ZL] have been based on finding copies of substrings, but for data compression, one copy is sufficient, as opposed to this work, in which the goal is to find all sufficiently long copies.

Exact Matching

This section describes how **dup** finds the set of pairs that are longest exact matches.

First, a lexical analysis phase hashes the lines in order to assign each line an identifying integer such that two lines are the same if and only if their ids are the same. The output from the lexical analysis phase can be considered a string of symbols over an alphabet of integers.

Given this string, a brute force method of finding longest matches would be to do the equivalent of scanning along diagonals of a scatter plot for longest matches, using time $O(n^2)$ and space $O(n)$, where n is the number of lines of input. No algorithm can avoid quadratic behavior in the worst case, since the number of longest exact matches can be quadratic in the size of the input. However, an exact matching algorithm need not exhibit quadratic behavior for every input.

In particular, two algorithms have been implemented, with a time-space tradeoff. The first algorithm runs in time $O(n+p)$, where p is the number of distinct pairs of identical lines. (For typical C code, p is dominated by the closing

braces that usually appear alone on a line.) The second algorithm, based on the suffix tree data structure [McC], runs much faster; on a fixed alphabet, it runs in time $O(n+m)$, where m is the number of matches found (ordinarily small compared to the size of the input). In practice, the alphabet is very large, but hashing is used to avoid a blowup in running time. Both algorithms use space linear in the size of the input, but the suffix-tree-based algorithm currently consumes three times as much space as the slower one, although further tuning may bring this down. Since the space consumption for millions of lines of code may be in the vicinity of main memory sizes, both algorithms are potentially of interest.

The first, more space-efficient, algorithm creates an array in which all identical lines are linked in lists, from the end of the file toward the front. Starting at the last line of the file, and working through the preceding lines, it "grows" matches through the file by following the linked lists and accumulating longer and longer matches. Two additional arrays are needed to keep track of the current lengths and the previous lengths computed. Thus, the algorithm requires three words per line of input.

The suffix-tree-based algorithm is more interesting. A suffix tree is a multiway Patricia trie; an example is given in Figure 3. Suppose the input is $b_1 b_2 \dots b_n$; without loss of generality, we assume that the last symbol, b_n , occurs only once in the input. Each leaf of the suffix tree represents a distinct suffix of the input, where a suffix is a substring $b_i \dots b_n$ for some i . Thus, there are exactly n leaves. Each arc of the tree is labelled with a nonempty substring of the input; the sequence of arcs from the root to a leaf yields the suffix represented by the leaf. For readability, in Figure 3, each leaf is labelled with the position of the start of the corresponding suffix and the substring representing the suffix, and each branching node is labelled with the sequence of labels on the path from the root to the branching node. Each node other than a leaf has at least two children; hence the number of nodes is linear in n . To make the size of the whole tree linear in the size of the input, the arc labels are stored as pointers into the input string. (Node labels need not be stored.) A suffix tree can be built in time linear in the size of the input, for a fixed alphabet [McC].

Now, two substrings of the input are a longest match if they are identical, but the preceding characters are different and the following characters are different. If two leaves are reached by different branches from a common ancestor B , their suffixes agree on the path from the root to B and then diverge. For example, the suffixes *bc%* and *bcbcab%* agree on the initial *bc*, which is the label on the edge from the root to their lowest common ancestor, labelled *bc*. Thus, the labels on the path from the root to a

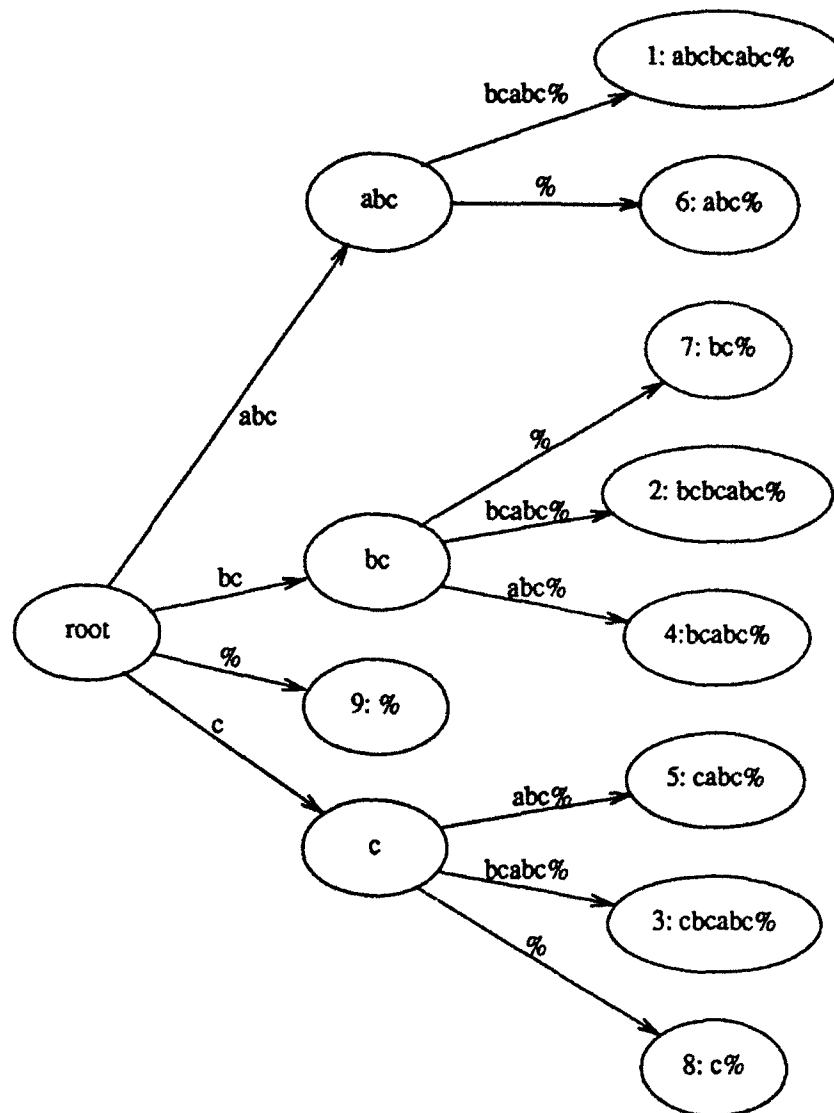


Fig 3. A suffix tree for the string *abcbcab%*.

branching node *B* in the suffix tree represent a substring that occurs in at least two places in the input, and their *right contexts*, i.e. the characters following the two occurrences, are different. However, the *left contexts*, i.e. the characters preceding the two occurrences, may be the same. Therefore, the program must do more work to determine which such pairs also have distinct left contexts, resulting in a longest match.

To do this, the program recurses over the suffix tree. It has two jobs to do: to build up lists of suffixes grouped by left context, and to compare the lists found for its subtrees to identify longest matches. Thus, for a node *N* and a symbol

c, a list $c:p_1, p_2, \dots, p_k$ will be created, where p_1, p_2, \dots, p_k are the starting positions for suffixes with left context *c* that correspond to leaves in the subtree for *N*. Identification of longest matches is by checking each pair of elements in the cross-product of each pair of lists from distinct subtrees with distinct left contexts. For example, at the branching node corresponding to *bc*, the occurrence of *bc* starting at position 4 (with left context *c*) has a longest match with the occurrences of *bc* starting at positions 2 and 7, because each of them has left context *a*. In order to get the desired $O(n+m)$ running time, the above computations are only performed for branching nodes representing substrings that are long enough to be reportable.

Parameterized Matching.

The program to find parameterized matches has three phases. First, the lexical analyzer copies each line while replacing each token name that is a candidate for a parameter into a P, while creating a separate list of all such token names. For example, $x=3*y;$ is turned into $P=P*P;$. Next, the transformed lines are passed to one of the exact matching algorithms described in the previous section. Exact matches that are sufficiently long are then analyzed for parameterized matches.

In such an exact match of transformed lines, the corresponding lines match exactly, including the P's that replaced parameter candidates. Therefore, the i th lines of the two segments must have the same number of parameter candidates in the same positions, for all i . If the original lines of the entire exact match have a parameterized match, then a one-to-one correspondence can be found that pairs the j th parameter name in the i th line of the first code segment with the j th parameter name in the i th line of the second code segment, for all i and j .

In general, it will not be possible to find a one-to-one correspondence for the entire exact match. Nevertheless, there may be parts of the exact match that have a sufficiently long parameterized match to report. For example, consider the following two code fragments.

```
x=y-z;
if (y>z)      m=1;
h=f(x);
y=x;

x=b-c;
if (b>c)      n=1;
h=f(x);
c=x;
```

Pairings $y=b$, $z=c$, and $m=n$ (and the identity on other parameter candidates) yield a parameterized match for the first four lines. However, the fifth line requires a pairing $y=c$, which conflicts with both $y=b$ and $z=c$. Pairings $y=c$ and $m=n$ (and the identity on other parameter candidates) yield a parameterized match for the last three lines.

Therefore, the algorithm scans the exact match line by line, keeping track of the start of the current match. If a conflict occurs between the one-to-one correspondence needed for a new pair of lines and the one-to-one correspondence that has already been established, then the current parameterized match is terminated, and if it is sufficiently long, reported as a longest parameterized match. The conflicting pairs are removed from the old one-to-one

correspondence, the pairs from the new line are added to it, and the line after the last conflict with the new one-to-one correspondence is taken as the beginning of a new parameterized match. When all the lines have been processed, the last parameterized match is reported if it is sufficiently long. The algorithm runs in time linear in the length of the exact match.

Discussion

Dup was implemented in about 2500 lines of C and lex [LS], and runs under *UNIX*TM. It has been applied to the source for the X Window System [SG] (minus some table initializations) and part of a larger AT&T software system. The parameterized matches for the X source and some of the AT&T system are plotted in Figures 4 and 5, respectively, where the minimum match length is 30 lines.

The parameterized matches include a substantial amount of the code in each case, namely 15% (1136 matches) for the X system, and 23% (596 matches), for the AT&T system. These numbers increase dramatically for smaller minimum match lengths; for example, if the minimum match length is reduced to 15 lines, the number of matches for the AT&T system increases to 4174 and the percentage of lines involved increases to 38%.

The plots are dense near the main diagonal, implying that most copies tend to occur fairly locally, e.g. within the same file or module. However, certain line segments occur away from the main diagonal; it would be interesting to investigate why the corresponding sections of code are duplicated.

When individual matches are examined, the matches for C code usually look reasonable. In particular, the one-to-one correspondence between parameters usually finds pairings of similar tokens, i.e. small integers with small integers or variables with other variables with related names.

When conflicts are found by the algorithm that generates parameterized matches from the exact matches for transformed lines, the conflicts frequently involve small integer constants, especially zero, used differently in two places.

There are two situations in which the program has produced output that did not appear reasonable. One situation is the initializations of large tables, with one value per line, where many matches may be found even for random values. These should probably be filtered out automatically. The other situation involves case statements, where a succession of statements of the form *case name:* allow a parameterized match to be found for a long list of probably unrelated cases. A reasonable solution

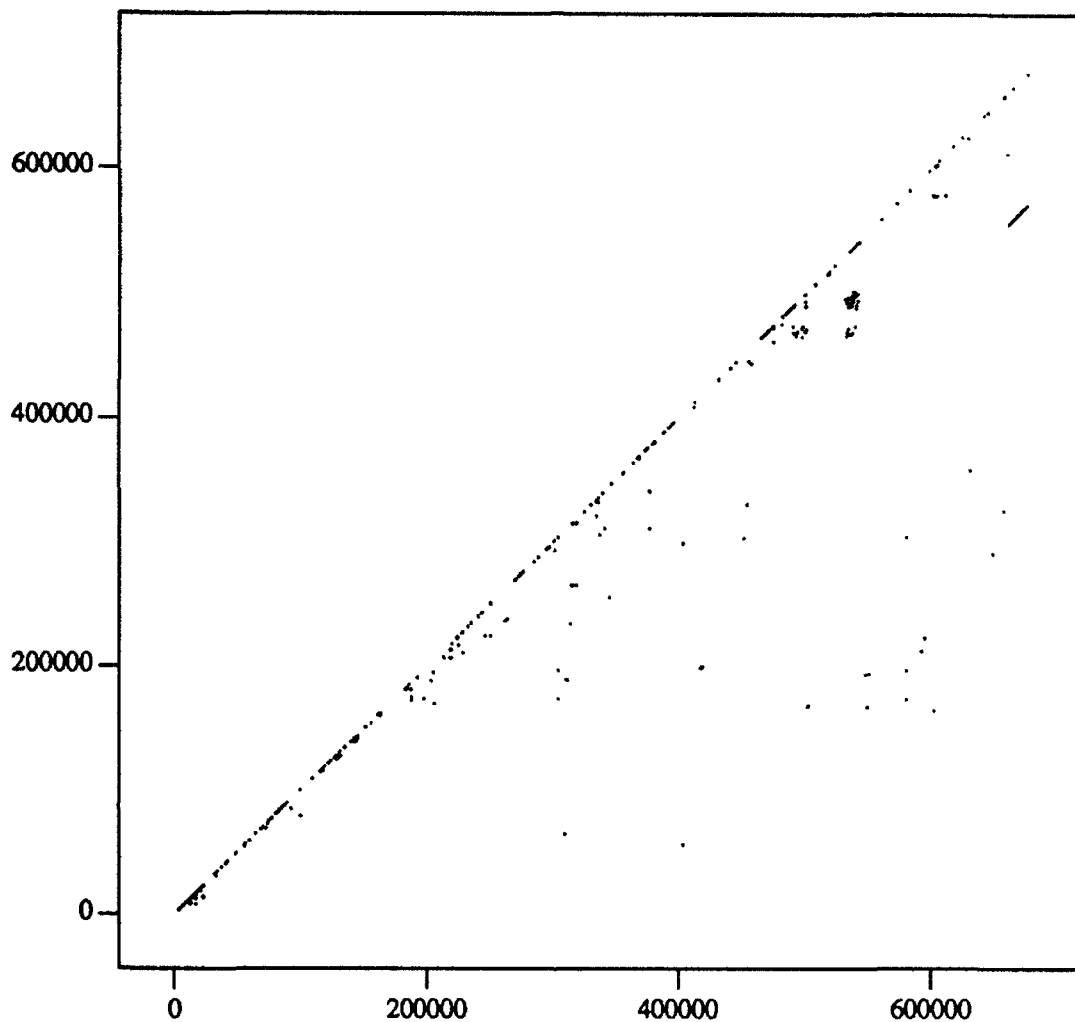


Fig. 4. Parameterized matches for X source (minus some tables).

for avoiding such output was found to be to limit parameterized matches so that the number of parameters is restricted to be at most half the number of lines.

In practice, many of the parameters found are related to error checking and handling, in both of the systems that were studied. The existence of unique constants used for error handling is one reason that many more parameterized matches are found than exact matches.

An obvious question is whether the output could be used to generate parameterized procedures automatically from the input to reduce the size of the code. Regretfully, the code segments identified in matches usually do not correspond exactly to subtrees in the parse tree for the program, or to any obvious semantic unit. A programmer would need to rewrite most files by hand, although the output from *dup* would undoubtedly be helpful.

However, the plots of large amounts of code should be useful to managers in visualizing the complexity and interrelationships of a whole software system. This paper includes plots of hundreds of thousands of lines of code. However, the program has been applied to a subsystem of over a million lines of code, and could be applied to still larger amounts of code.

Moreover, zooming in on smaller sections can be useful for finding anomalies in the scatter plot that reflect anomalies in the code. Three types of interesting features that have been found in casual browsing have been unusually complex files, an obsolete file, and a place where a bug fix was apparently applied to one copy of some code but not to another other copy. The obsolete file was found by noticing rather extensive duplication between two files in a module. Figure 6 shows a scatter plot with a gap between

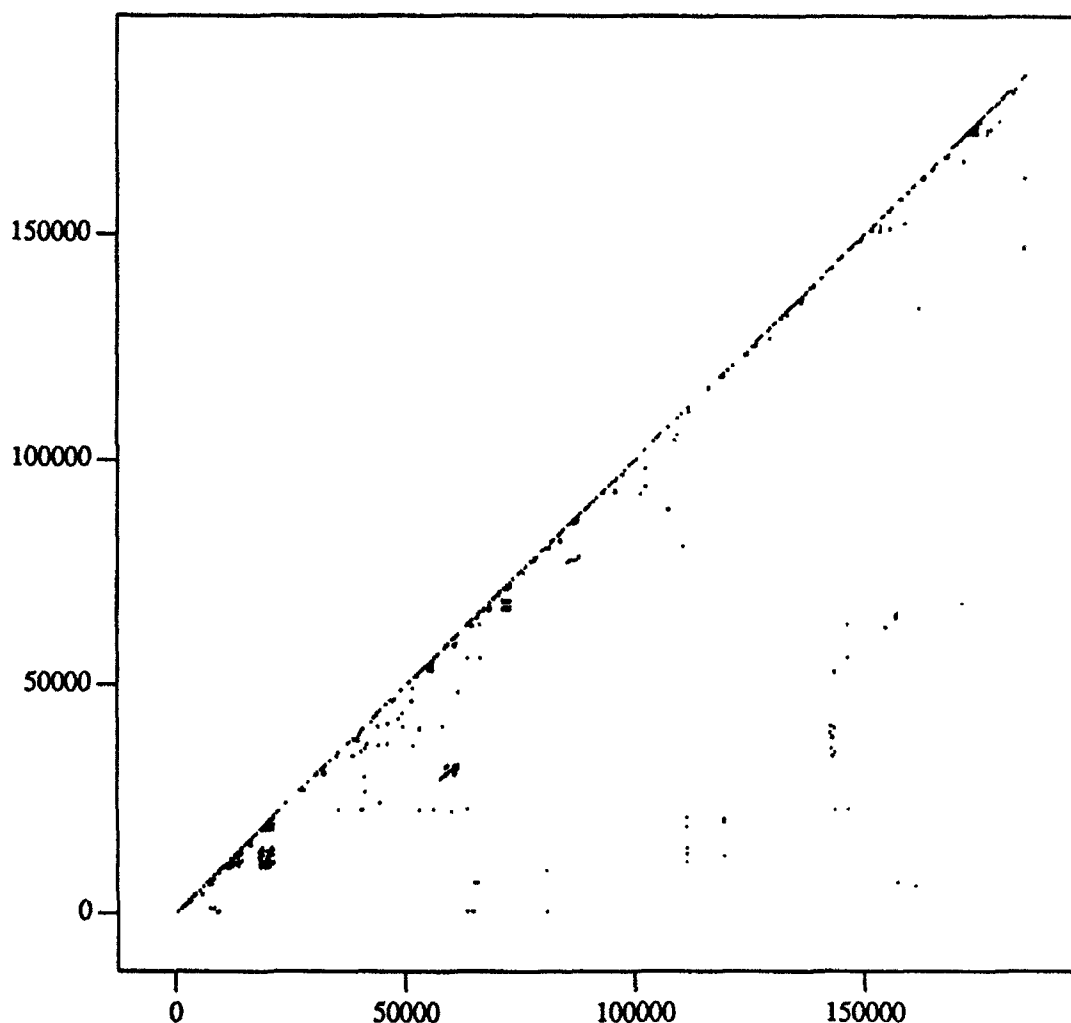


Fig. 5. Parameterized matches for some code from a production system.

two collinear line segments representing two long matches. The gap corresponds to two sections of code, one of which is a loop that runs off the end of an array, and the other of which has a correct loop with a comment describing the correction; the latter is apparently a bug fix that was applied in one copy and not the other.

Acknowledgements

I would like to thank Brian Kernighan for calling my attention to the code duplication problem and for suggesting the first lexical analysis part of the parameterization algorithm. I would also like to thank William Chang for providing his code for suffix tree construction. I am appreciative of many useful discussions with Ken Church, Ken Clarkson, Bryan Ewbank, Raffaele Giancarlo, Eric

Grosse, Jon Helfman, Andrew Hume, Brian Kernighan, and Eric Sumner.

References

- [CH] Kenneth W. Church and Jonathan I. Helfman, Dotplot: a program for exploring self-similarity in millions of lines of text and code, *Computing Science and Statistics: Proceedings of the 24th Symposium on the Interface* (1992).
- [E] Stephen G. Eick, Dynamic Graphics for Software Visualization, *Computing Science and Statistics: Proceedings of the 24th Symposium on the Interface* (1992).
- [H] M.H. Halstead, *Elements of Software Science*, Elsevier North-Holland, New York (1977).

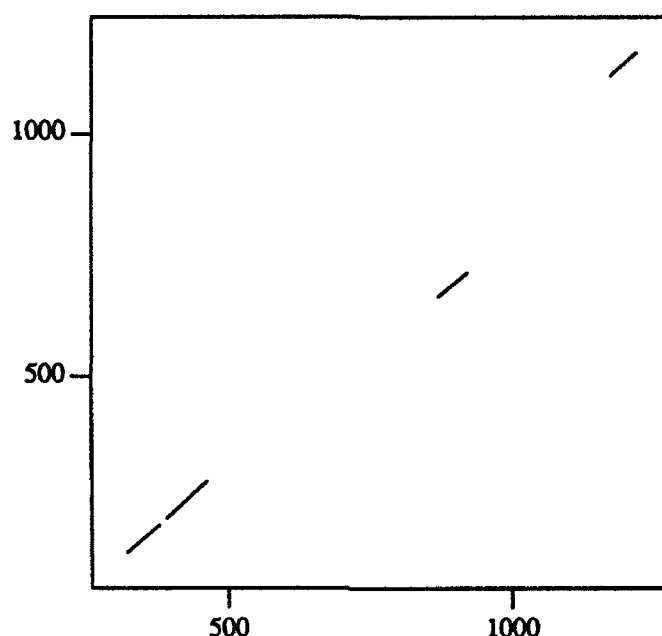


Fig. 6. A plot with an anomaly: a small gap between two matches.

- | | | | |
|-------|---|-------|---|
| [Ja] | H.T. Jankowitz, Detecting plagiarism in student PASCAL programs, <i>Computer Journal</i> 31,1 (1988), pp. 1-8. | [McC] | E.M. McCreight, A space-economical suffix-tree construction algorithm, <i>J. ACM</i> 23,2 (1976), pp. 262-272. |
| [Jo] | Ralph Johnson, personal communication (October, 1991). | [RPE] | M. Rodeh, V. R. Pratt, and S. Even, Linear algorithms for data compression via string matching, <i>J. ACM</i> 28,1 (1981), pp. 16-24. |
| [KP] | Brian W. Kernighan and Rob Pike, <i>The UNIX Programming Environment</i> , Prentice-Hall (1984), Englewood Cliffs, New Jersey. | [SK] | D. Sankoff and J.B. Kruskal (editors), <i>Time Warps, String Edits, and Macromolecules: the Theory and Practice of Sequence Comparison</i> (1983), Addison-Wesley, Reading, MA. |
| [LS] | M.E. Lesk and E. Schmidt, Lex- a lexical analyzer generator, <i>UNIX Programmer's Manual</i> , Holt Reinhart and Winston (1983-1984). | [SG] | R.W. Scheifler and J. Gettys, The X window system, <i>ACM Transactions on Graphics</i> 5,2 (1986), pp. 79-109. |
| [LMW] | Richard J. Lipton, Thomas G. Marr, and J. Douglas Welsh, Computational approaches to discovering semantics in molecular biology, <i>Proc. of the IEEE</i> 77,7 (1989), pp. 1056-1060. | [ZL] | J. Ziv and A. Lempel, A universal algorithm for sequential data compression, <i>IEEE Trans. Inf. Theory</i> IT-23 (1977), pp. 337-343. |

Dotplot: a Program for Exploring Self-Similarity in Millions of Lines of Text and Code

Kenneth Ward Church
AT&T Bell Laboratories
Murray Hill, NJ 07974-2070
kwc@research.att.com

Jonathan Isaac Helfman
AT&T Bell Laboratories
Murray Hill, NJ 07974-2070
jon@research.att.com

ABSTRACT

An interactive program, *dotplot*, has been developed for browsing millions of lines of text and source code, using an approach borrowed from biology for studying homology (self-similarity) in DNA sequences. With conventional browsing tools such as a screen editor, it is difficult to identify structures that are too big to fit on the screen. In contrast, with dotplots we find that many of these structures show up as diagonals, squares, textures and other visually recognizable *features*, as will be illustrated in examples selected from biology and two new application domains: text (AP news, Canadian Hansards) and source code (SESS®). In an attempt to isolate the mechanisms that produce these features, we have synthesized similar features in dotplots of artificial sequences. We also introduce an approximation that makes the calculation of dotplots practical for use in an interactive browser.

1. Introduction

Most work in the interface between computer science and statistics uses computational techniques in support of statistics. This paper attempts the reverse. We describe a graphical tool for browsing millions of lines of text and source code. It is hard to use a screen editor to conceptualize input that is much larger than the size of a screen. Following Eick (1992), who advocates the use of interactive graphical tools to help understand large software systems, we have developed a browser that can display millions of lines of input using a *dotplot*, a plot very much like those used in molecular biology for studying homology (not to be confused with Tukey's "dot plot" (1977, p.50)). We believe the tool may be useful for discovering large-scale structures that may be hard to spot with conventional tools such as a screen editor: conventional tools may be too myopic to show the big picture.

Fig. 1 shows the browser in action. Three views of a source code file are presented: a) a global overview of the file in the upper right, b) a magnified view of a small portion of the file in the upper left, and c) a text view along the bottom. The views are linked together so that clicking and scrolling in one view updates the others appropriately.

Notice the fascinating *diagonals*, *squares*, and *textures* in Fig. 1. The texture labeled D will be discussed in more detail in Section 4.2. What mechanisms could be responsible for these *features*? What do the features tell us about the input sequence? This paper uses two approaches to investigate such questions. In addition to the browser, which allows us to analyze naturally occurring sequences, we also synthesize artificial sequences in an attempt to replicate features found with the browser. Both methods, analysis and synthesis, are used to study the mechanisms that might be responsible for the features.

Fig. 2 shows several synthesized dotplots. Fig. 2a, for example, was generated from the artificial sequence: "zyxwvutsrqponmlkji". A dot is placed in position i, j if the i^{th} input token is the same as the j^{th} . In this case, dots appear along the main diagonal and nowhere else, because all of the input tokens are distinct. (Since the main diagonal is uninteresting, it will be omitted from subsequent

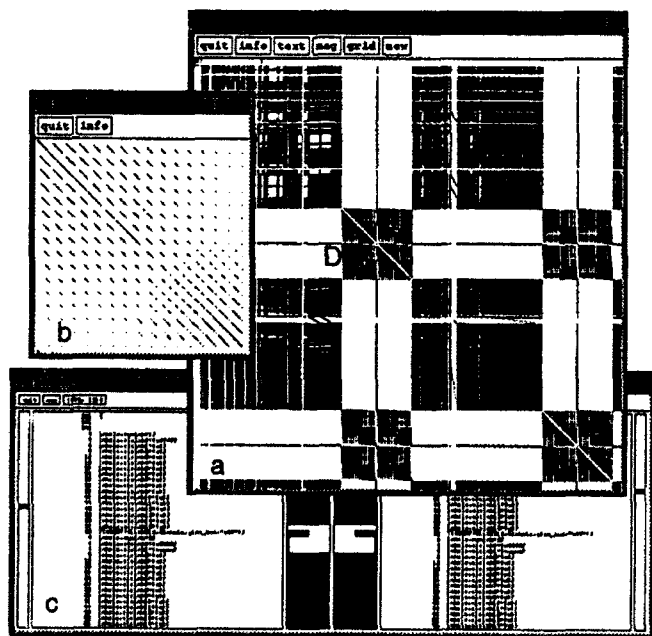


Fig. 1. Dotplot Browser

dotplots.) In contrast with Fig 2a, there are two interesting diagonals in Fig. 2b, indicating that the subsequence "abcdefghi" is repeated. We have found that diagonals, and other features, are often symptomatic of certain potentially important patterns in the input sequence.

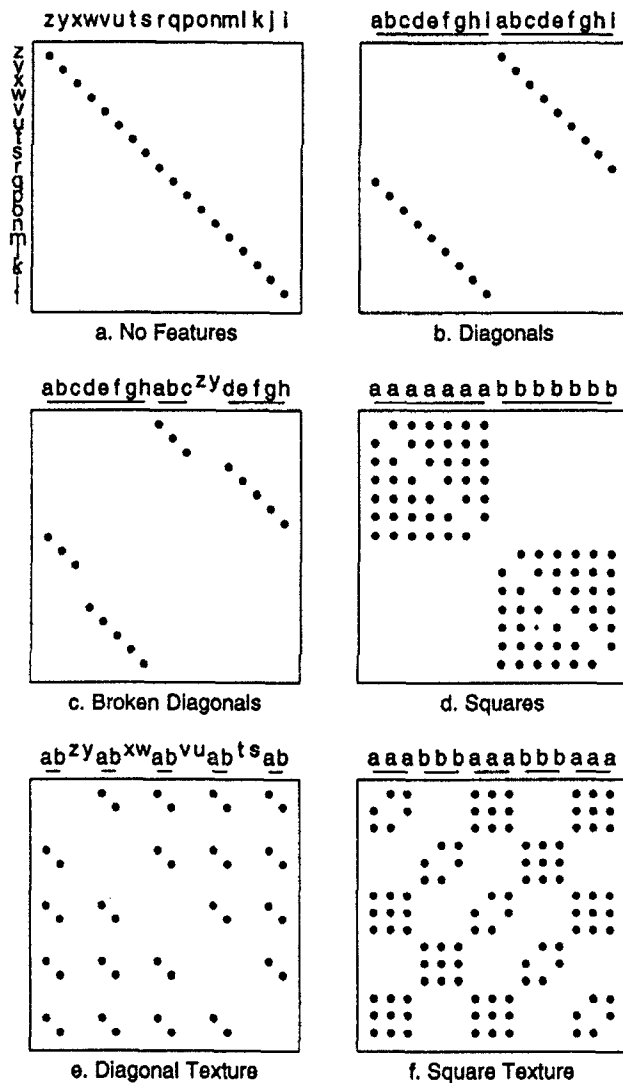


Fig. 2. Features in Synthesized Dotplots

A number of conventions will be used throughout this paper. Underlining (as in Fig. 2b) is used to emphasize tokens that repeat; raising the baseline (as in Fig 2c) is used to emphasize tokens that do not repeat. In general, letters from the beginning of the alphabet are used to denote repeating tokens, and letters from the end of the alphabet are used to denote non-repeating tokens. In order to save space, labels along the left margin are often omitted. Finally, the somewhat unusual convention of placing the origin in the upper left corner was chosen in order to conform with the fact that English text is read left to right and top to bottom. The interaction of the text views and dotplot views is more natural when the location of the origin is consistent.

2. Dotplots of DNA Sequences

The features in Figs. 2b-f can also be found in dotplots of real sequences. For example, diagonals are found in Fig. 3, a dotplot of two concatenated DNA sequences: (A) the plasmid pBR322 (Balbas et al. 1986), and (B), the plasmid pUC19 (Yanisch-Perron et al. 1985). Dotplots are a well-known technique in biology for studying homology (e.g., Maizel & Lenk 1981, Pustell & Kafatos 1982). Biologists are very interested in diagonals which indicate, in this case, that both pBR322 and pUC19 carry the β -lactamase gene that confers ampicillin resistance. Biologists have also used dotplots to look at how sequences fold into three-dimensional structures (e.g., Quax-Jeuken et al. 1983, Blundell et al. 1987, Carrington & Morris 1987), and to investigate evolutionary questions (e.g., Laver et al. 1980, Doolittle 1981, Lake et al. 1988). A brief description of the value of this approach can be found in Argos (1987). See Vingron (1991) for a recent thesis on genetic sequence alignment.

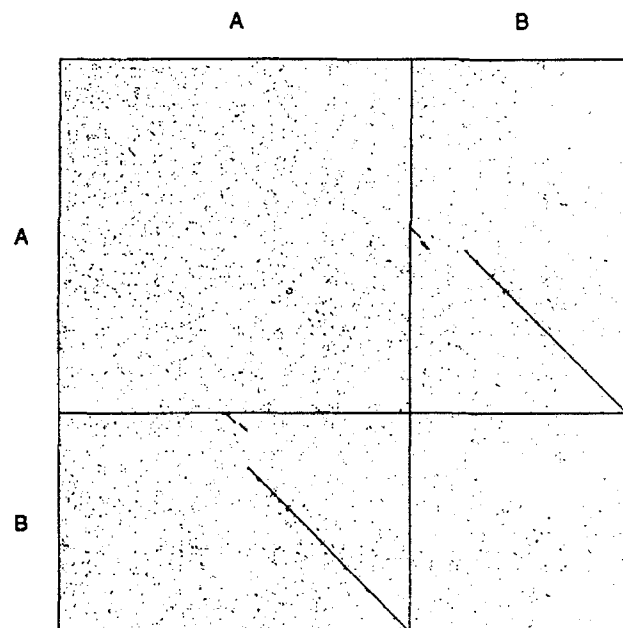


Fig. 3. Dotplot of Two DNA Sequences (7000 Nucleotides)

A few additional conventions are introduced in Fig. 3, and will be used throughout this paper. First, grid lines are used to indicate the boundaries between sequence A and sequence B. In addition, the grid box in the upper left corner will be called "AA," the grid box in the upper right will be called "AB," and so on. Grid box AA compares sequence A with itself, while grid box AB compares sequence A with sequence B. In general, the grid boxes along the main diagonal compare two identical sequences, while the other grid boxes compare a pair of different sequences.

Note that the diagonals are broken. What does this mean? Fig. 2c (above) shows, by synthesis, that broken diagonals

are caused by the insertion of non-repeating tokens (zy) into an otherwise matching subsequence (abcde~~fgh~~). In the case of Fig. 3, the breaks probably indicate that some non-repeating nucleotides have been inserted near the beginning of B, interrupting the match between the second half of A and most of B.

3. Dotplots of Text

3.1 AP News: Broken Diagonals in Text

Broken diagonals can also be found in dotplots of text, as illustrated in Fig. 4, a dotplot of four Associated Press (AP) news stories, labeled A-D. The four stories, about Ryan White's death from AIDS, were sent over the AP wire within a few days of each other in early April of 1990. For text applications, we usually choose to tokenize the input into words.

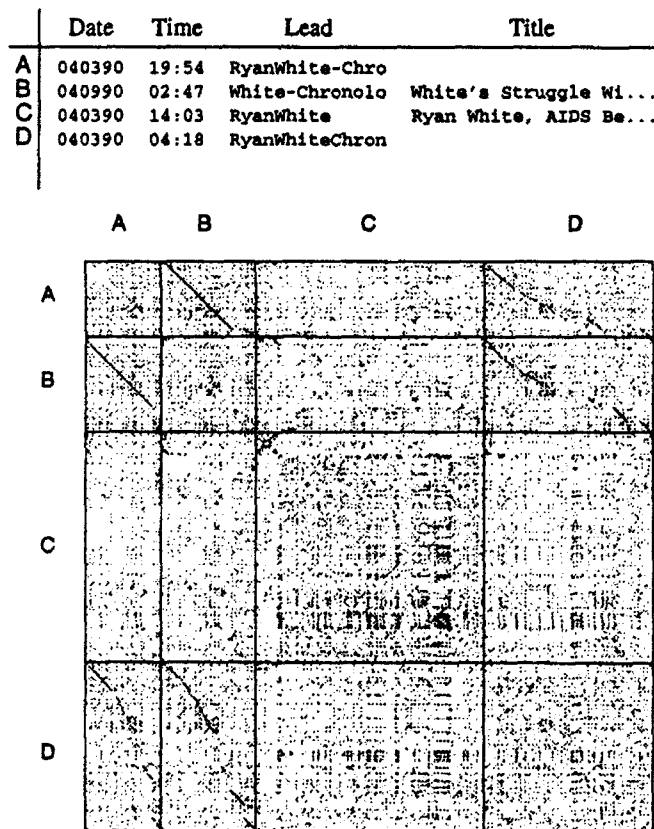


Fig. 4. Four AP News Stories (3000 Words)

What do the broken diagonals tell us about these AP news stories? We suspect that the diagonals are broken when a news story is updated with a few additional facts. Stories A, B, and D appear to be related in this way, as evidenced by the broken diagonals in grid boxes: AB, BA, AD, DA, BD, DB. In contrast, story C is probably not a rewrite of the others, as evidenced by the absence of the broken diagonals in: AC, CA, BC, CB, DC, CD.

Dotplots may have practical ramifications for Information Retrieval (IR) (Salton 1989). There are a number of IR systems that provide rapid access to documents in large electronic libraries. Experience with such systems has shown that users find it difficult to construct queries that cover most of the documents of interest and not too many others. This problem could be alleviated, in part, if document retrieval systems had a more effective way of handling rewrites. In particular, the user is probably only interested in one of the rewrites, e.g. story D. To date, most retrieval systems consider each document one at a time, and consequently, they would usually return either all of the rewrites or none of them. There is generally no easy way to retrieve just one of the rewrites, along with an indication that there are a few more that are nearly the same.

Thus far we have seen two examples of broken diagonals. The next section shows how diagonals can be combined with squares.

3.2 Hansards: Combinations of Sparse Features

Fig. 5 is a dotplot of 37 million words of Canadian *Hansards*, parliamentary debates, which are available in both English and French. The input is constructed by concatenating 3 years of debates in English (37/2 million words) followed by the French equivalent (the remaining 37/2 million words). Consequently, there is a lag of approximately 37/2 million words between an English sentence and its French translation.

Note the diagonals and large dark squares in Fig. 5. We have seen examples of these features in isolation, but what mechanism could explain the combination? Figs. 6 and 7 present a two step solution: first, sparse versions of the diagonals and squares are synthesized, as illustrated in figs. 6b and 6d, and then the interesting halves of Figs. 6b and 6d are interleaved to produce the desired combination in Fig. 7.

How does the synthetic sequence in Fig. 7 relate to the *Hansards*? Let the a's denote English words, e.g. *government*, the b's denote French words, e.g. *gouvernement*, and the c's denote words that are the same in both English and French, e.g. proper nouns, dates, times, numbers, etc. We hypothesize that the square in the upper left is formed because there are many a's matching a's or English words matching English words. Similarly, the square in the lower right is probably formed because there are many b's matching b's or French words matching French words. We suspect that the diagonals indicate how the English text should be aligned with the French. There is a good chance of a dot contributing to the diagonal when the two texts are so aligned because there are a fair number of proper nouns, dates, times, numbers, etc. that will match when text is compared with its translation. There is also an additional pattern in the input that is responsible for the main diagonal.

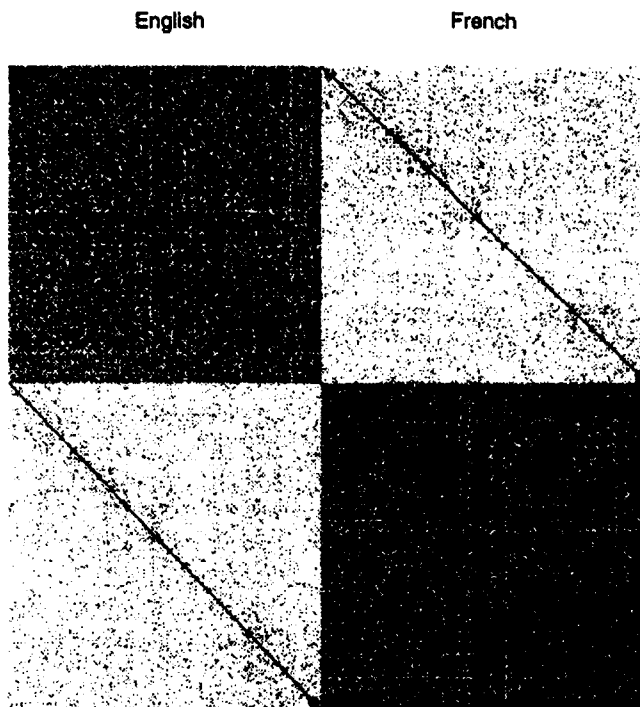


Fig. 5 Three Years of Hansards (37 Million Words)

a^ca^da^ea^fa^ga^haⁱa^ja^kab^cb^db^eb^fb^gb^hbⁱb^jb^kb

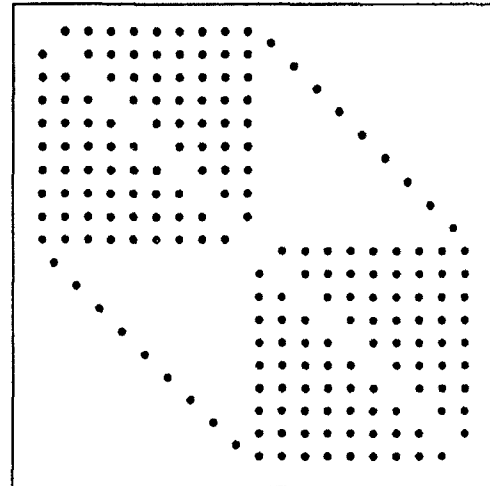
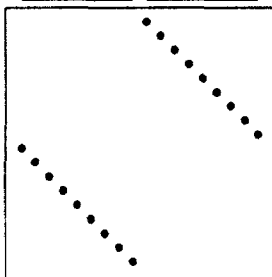


Fig. 7. Combination of Diagonals and Squares

Of all the dotplots in this paper, Fig. 5 has the highest *data-ink ratio*. (Tufte 1983, p.93). 37 million words is at least four orders of magnitude more than could be seen with a conventional text editor. The tremendous compression factor of Fig. 5 increases the apparent density of features that are actually quite sparse.

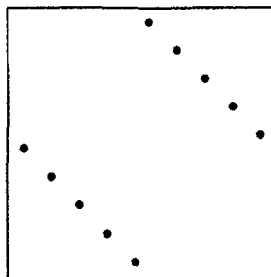
To summarize, we have seen two examples of text dotplots, AP news and Hansards, and two types of features, diagonals and squares. Diagonals indicate matches and alignments, while squares indicate regions of local similarity. We have also seen how features can be preserved despite changes in density, and how this property allows us to interleave combinations of sparse features. These ideas will be further developed in the next section which introduces the source code application.

abcdefghiabcdefghi



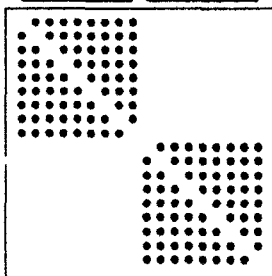
a. Dense Diagonals

a^zb^yc^xd^we^av^bu^ct^ds^e



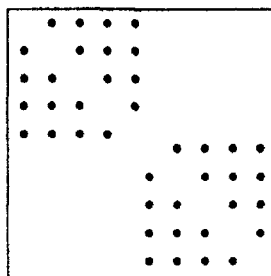
b. Sparse Diagonals

aaaaaaaaabbbbbbbbbb



c. Dense Squares

a^za^ya^xa^wab^vb^ub^tb^sb



d. Sparse Squares

Fig. 6. Dense vs. Sparse Features

4. Dotplots of Source Code

4.1 Diagonals in Source Code

The source code examples in this paper are taken from the 5ESS® switch, a large program that handles much of the world's long distance telephone service. For source code applications, we usually choose to tokenize the input into lines of code.

Fig. 8 shows a dotplot of eight source code files labeled A-H. Two features are of interest: (1) a long broken diagonal starting at AE and extending down to DH (and, by symmetry, another long broken diagonal starting at EA and extending down to HD), and (2) 56 short diagonals, each starting in the upper left corner of a different grid box.

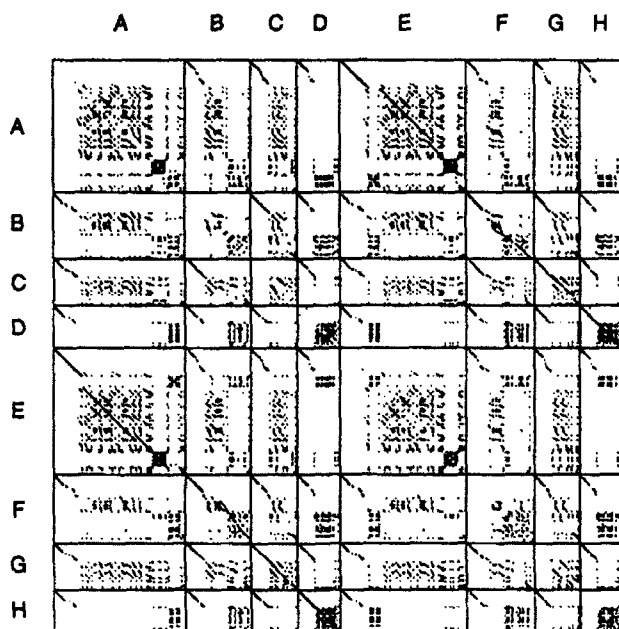


Fig. 8. 3000 Lines of Code

What could cause these features? The first feature, the long broken diagonals, indicates that files A, B, C, and D are similar to files E, F, G, and H, respectively. This observation is further supported by the striking pattern in the names of the files, as shown below. Perhaps these files were copied to maintain a parallel version of the software.

First 4 Files		Second 4 Files	
A	P.ISqf3.c	E	P.ISqf4.c
B	P.ISqf3_hold.c	F	P.ISqf4_hold.c
C	P.ISqf3_hr.c	G	P.ISqf4_hr.c
D	P.ISqf3_rr.c	H	P.ISqf4_rr.c

The second feature, the 56 short diagonals, has a different explanation. Each of the eight files starts with a highly structured comment of the form:

```
/*
 * File:    ...
 *
 * Data:    ...
 *
 * Name:    ...
```

The comments also include a number of additional fields: Abstract, Loadable Package, Usage, Parameters, Externals, etc. We believe the 56 short diagonals are caused by similarities in the eight comments.

4.2 Textures in Source Code

A relatively small number of comments in Fig. 8 generate a relatively large number of diagonals. In general, n copies of a subsequence generate $n(n-1)$ diagonals. Consequently, even a relatively small number of copies will generate such

a large number of diagonals that they form a texture. A good example can be found in Fig. 1a (near the label D), shown in more detail in Fig. 9.

This texture consists of a large number of diagonals of varying lengths. Consider the upper left corner where diagonals are shrinking. Fig. 10 shows, by synthesis, that diagonals shrink as a repeating subsequence is diluted with increasing numbers of non-repeating tokens.

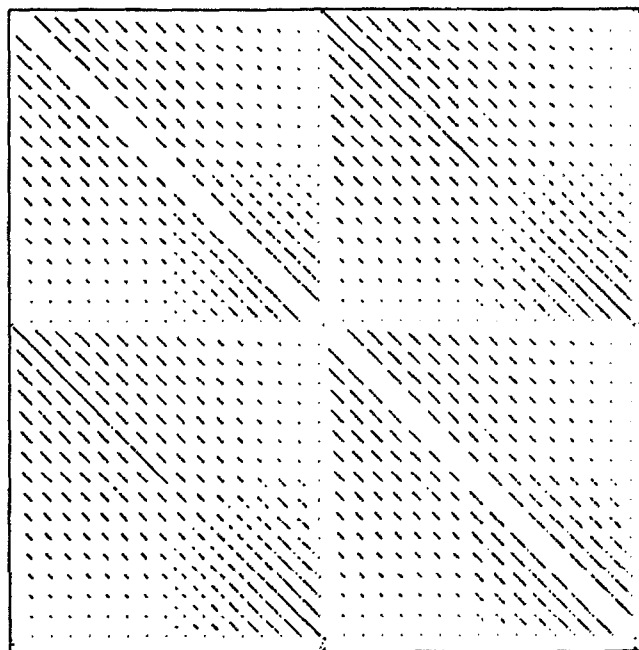


Fig. 9. 600 Lines of Code (Detail of Fig. 1)

abcdefghijklmnopqrstuvwxyzbcdefghijklmnopqrstuvwxyz

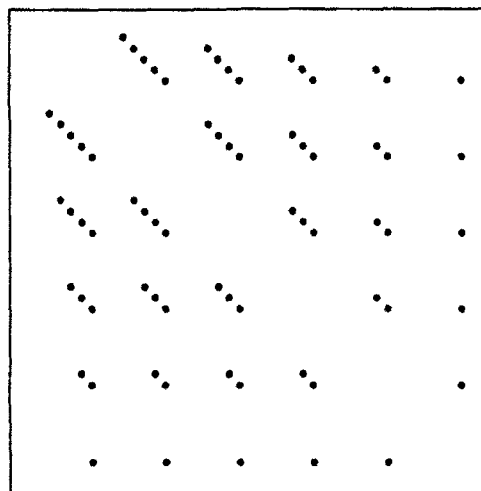


Fig. 10. Shrinking Diagonals

How does the artificial sequence in Fig. 10 relate to the actual input sequence for Fig. 9? We believe the texture in Fig. 9 is generated by two clauses of an *if* statement. Each clause consists of 16 groups of 18 lines of code. The first

group initializes all but the first field of a structure to 0; the second initializes all but the first two fields to 0; the third initializes all but the first three fields to 0; etc. The repeating sequence of initializations to 0 is diluted with increasing numbers of non-zero initializations resulting in the "shrinking diagonals" texture. There is also another pattern in the code that causes diagonals to grow longer toward the lower right corner of the texture.

This example shows that dotplots highlight a level of structure that would have been very difficult to discover using traditional tools, such as a screen editor, because the pattern extends over several hundred lines of code, much more than could possibly fit on a screen. In addition, this structure would also be difficult to appreciate with a dynamic programming approach such as the UNIXTM diff program. Such programs attempt to find a single match, and are therefore unable, in principle, to find the rich texture of multiple overlapping matches. In addition, the diff program would have trouble in this case because many of the matches aren't exact. To handle cases like this, Baker (1992) introduced an inexact matching criterion, *parameterized match*, which overcomes this difficulty by equating two lines that are the same up to the names of the parameters and the values of the constants. But unfortunately this equivalence relation would also miss the pattern of "shrinking diagonals," because it depends crucially on the names of the parameters and the values of the constants.

However, in other cases, equivalence relations have proved to be extremely powerful. Consider, Fig. 11, for example, where it appears that several large sections of code were copied verbatim (white space and all), as evidenced by the long diagonals. Suppose we wanted to understand more about the copied code: Who copied it? When? Why?

4.3 Attributes

Author Attributes	
Author	Code
carlson	/*
carlson	• Name: RTgeninit
kedzierski	•
veach	#feature (5E2_2G)
martin	• Module: RTmain
martin	•
ahmad	#endfeature (5E2_2G)

One approach to answering these kinds of questions makes use of an equivalence relation we call *attributes*. Large software development projects typically maintain a database that associates each line of code with various attributes such as the author's name, modification dates, etc. The table above illustrates the author attributes for the first few lines of code that were used to generate Fig. 11.

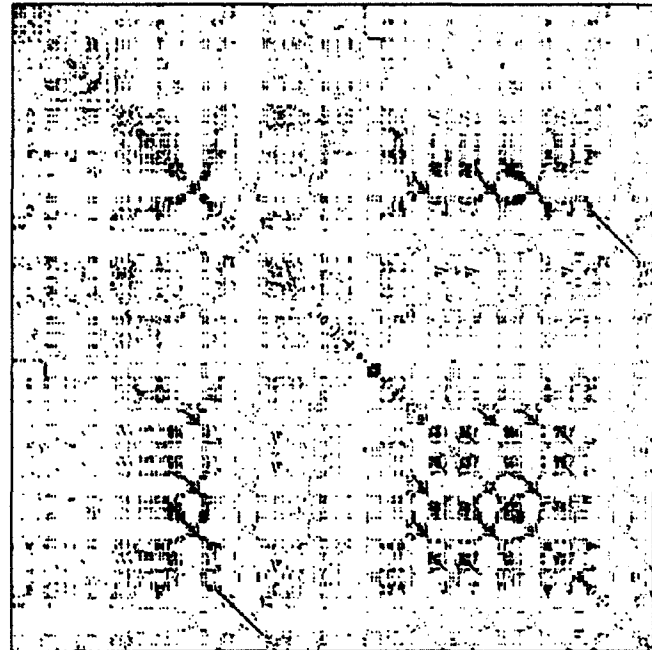


Fig. 11. 3400 Lines of Code

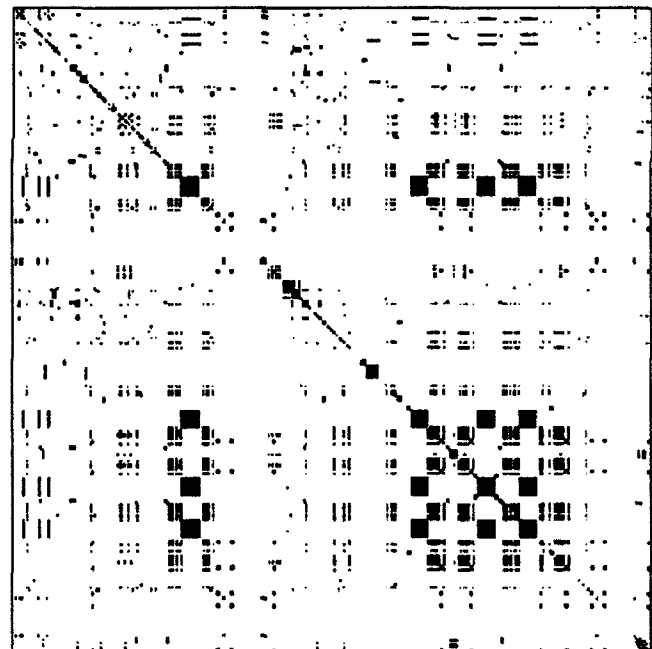


Fig. 12. 3400 Author Attributes

A dotplot can be generated from any input. Fig. 12, for example, is just like Fig. 11 except that it uses the author attributes (column 1) as input instead of the code (column 2). It is interesting to compare Figs. 11 and 12 in order to see if there might be a pattern between the code and the authors. In this case, at least, it appears that many of the diagonals in Fig. 11 coincide with squares in Fig. 12.¹

1. This observation will be easier to see in a forthcoming version of the browser, which will enable users to overlay multiple dotplots.

Why? We suspect that the copies were created by the same author, at the same time, and for the same reason. This conjecture can be further tested by looking at a dotplot of modification dates, and seeing if the features in that dotplot line up with those in Figs. 11 and 12.

4.4 Summary

In summary, we have explored the properties of a variety of features, primarily diagonals, but also squares and textures (Fig. 2). We have seen that features can appear in many variations: they can be broken (Fig. 2c), dense or sparse (Fig. 6), and they can appear in various combinations (Fig. 7). We have also seen applications of dotplots in biology (Fig. 3), as well as the two new applications: text (Fig. 4-7) and source code (Fig. 8-12). Both analysis and synthesis have been used to learn more about the relationship between features and corresponding patterns in input sequences.

Dotplots may have a number of practical ramifications for source code applications. First, dotplots might be useful for identifying large structures in a program, especially during *discovery*, the process of reading code for the first time. Secondly, dotplots might help developers find undesirable duplication so that it can be removed. In some cases, for example, it is possible to replace multiple copies with a single subroutine, as suggested in Baker (1992). In other cases, the ability to identify multiple copies can be useful for maintenance. In particular, if a bug is found in one of the copies, then there is a good chance that the others might require attention, as well. Thus, dotplots appear to be useful for identifying large structures, removing undesirable duplication when possible, and coping more effectively with duplication that cannot be removed.

Some users might believe that redundancy is always indicative of a weakness of some kind. For example, one user has started using the browser to identify C constructions, such as `switch` statements, which are often associated with a texture generated by repeated `break` statements. In this way, dotplots have been used to help design a new programming language that avoids many of these "wordy" constructions.

Should redundancy be considered "harmful"? Following a policy like Dijkstra's stand on `gotos`, one might suggest that redundancy should be eliminated, whenever possible. Unfortunately, such a policy would also remove a number of very useful structures such as the structured comments in Fig. 8. Carried to its logical extreme, such a policy would reduce a structured program to a random string, a string whose shortest description is itself. As in good writing, repetition can be a powerful rhetorical device for conveying emphasis, parallelism, etc. It would be a mistake to discourage such practices in a futile attempt to eliminate "wordiness" and other forms of "bad" writing.

5. Software Design

The next three subsections describe the implementation of the browser. First, the input data is tokenized into a sequence of N tokens. Secondly, this sequence is used to construct the *f-image*, an array of floating point values. Finally, these values are quantized into the *q-image*, an array which is suitable for displaying on a color or grey-scale monitor.

tokens \rightarrow f-image \rightarrow q-image

5.1 Tokenization

The program begins by tokenizing the input and applying the appropriate equivalence relations, if any. Equivalence relations were discussed briefly in sections 4.2 and 4.3; they can be used to remove white space, simulate a parameterized match, replace a token with one of its attributes, etc. The details of the tokenizer depend on the particular application. In the text application, for example, we have tended to tokenize the input text into words, whereas in the source code application, we have tended to tokenize the input code into lines.

Before discussing the next topic, the calculation of the *f-image*, it might be worthwhile to clarify a potential source of confusion between the term *type* and the term *token*. Consider, for example, the English phrase, "to be or not to be," which contains 6 words, but only 4 of them are distinct. We say that the sentence contains 6 tokens, but only 4 types. By convention, we denote the number of tokens in the input data with the variable N , and we denote the number of types in the input data with the variable V (for "vocabulary size").

One might normally choose to represent types as strings. That is, it would be natural to represent the word, *to*, as the string "to", and the line of code, `for(i=1; i<N; i++)`, as the string "for(i=1; i<N; i++)". For computational convenience, we have decided not to represent types as strings, but rather as contiguous integers in the range of 0 to $V-1$. The strings are converted to numbers using standard hashing techniques. Representing tokens as integers has several advantages. In particular, it makes it easy to test whether or not the type of the i^{th} token is the same as the type of the j^{th} token: `if(tokens[i] == tokens[j])`. If we had used strings instead of integers, then we would have had to use `strcmp` instead of `==`, which would have been much less efficient.

5.2 Computing the F-image

After the input data has been parsed into a sequence of tokens, the tokens are then converted into a floating point image, the *f-image*. In the simplest case, this is

accomplished by placing a dot in `fimage[i][j]` if the type of the i^{th} token is the same as the type of the j^{th} token. In other words:

```
float fimage[N][N];

for(i=0; i<N; i++)
  for(j=0; j<N; j++)
    if(tokens[i] == tokens[j])
      fimage[i][j] = 1;
    else fimage[i][j] = 0;
```

In order to compute dotplots more quickly and effectively, we make use of three observations: (1) *Weighting*: tokens should be weighted to adjust for the fact that some matches are more interesting than others; (2) *Compression*: if N is large, it becomes impractical to allocate N^2 storage, and therefore it becomes necessary to compress the image in some way; (3) *Approximation*: if N is large, the time to compare all N^2 pairs of tokens also becomes impractical, and therefore it becomes necessary to introduce certain approximations.

5.2.1 Weighting

The calculation above can be improved by replacing "`fimage[i][j] = 1;`" with "`fimage[i][j] = weight(tokens[i]);`", where the function `weight` returns a value between 0 and 1, depending on how surprising it is to find that `tokens[i] == tokens[j]`. There are quite a number of reasonable functions to use for `weight`. The weighting concept is illustrated below, using the natural suggestion of weighting each match inversely by the frequency of the type. In this way, frequent types (e.g., the English word *the* or the line of C-code `"")` do not contribute very much to the f-image because matches among such frequent types are not very surprising. The literature on IR contains considerable discussion of weighing, where it is called *term weighting* or *indexing*. See Salton (1989) for a recent secondary source on the subject.

```
/* Initialize freq */
float freq[V] = {0};
for(i=0; i<N; i++)
  freq[tokens[i]]++;

for(i=0; i<N; i++)
  for(j=0; j<N; j++)
    if(tokens[i] == tokens[j])
      fimage[i][j] = 1/freq[tokens[i]];
    else fimage[i][j] = 0;
```

5.2.2 Compression

If N is large, it becomes impractical to allocate N^2 storage, and therefore it becomes necessary to compress the image in

some way. Suppose that we wanted to compress the f-image from N by N , down to n by n , for some $n \ll N$. Then we could simply aggregate values that fall into the same n by n cell as shown below. Of course, it is recommended that the signal be filtered appropriately before compression in order to avoid aliasing (Gonzalez & Wintz 1987, p.94). Filtering may also be useful if there are too many dots in the f-image as is well known in the biology application (Maizel & Lenk 1981, Pustell & Kafatos 1982). In general, various well-known signal processing techniques might be useful for enhancing features of interest.

```
/* Initialize f-image */
float fimage[n][n] = {0};
/* Map x from token coordinates
   into f-image coordinates */
#define CELL(x) (((x) * n) / N)

for(i=0; i<N; i++)
  for(j=0; j<N; j++)
    if(tokens[i] == tokens[j])
      fimage[CELL(i)][CELL(j)] +=
        weight([tokens[i]]);
```

5.2.3 Approximation

In practice, if N is very large, it becomes impractical to perform the N^2 comparisons and it is therefore useful to introduce an approximation. In particular, we assume that extremely frequent tokens will have vanishingly small weights, which can be approximated as zero. Consequently, it becomes unnecessary to compute their contributions to the f-image, producing a significant savings in time.

Before presenting the approximation, it is convenient to introduce the concept of a *posting*, a precomputed data structure that indicates where a particular type can be found in the input sequence. Thus, for the input sequence, "to be or not to be," there are two postings for the type "to": one at position 0 and the other at position 4. One can compute the dots for the type "to" in this example by placing a dot in positions: (0, 0), (0, 4), (4, 0), and (4, 4). In general, for a word with frequency f , there are f^2 combinations of postings that need to be considered. The algorithm below simply iterates through all f^2 combinations for each of the V types in the vocabulary.

```
for(type=0; type<V; type++) {
  w = weight(type);
  f = freq[type];
  postings = get_postings(type);
  for(p1=0; p1 < f; p1++) {
    i = postings[p1];
    for(p2=0; p2 < f; p2++) {
      j = postings[p2];
      fimage[CELL(i)][CELL(j)] += w;
    }
  }
}
```

We now come to the key approximation. If we assume that types with large frequencies ($f \geq T$, for some threshold T) have negligible weights, then we don't need to iterate over their postings. This approximation produces significant savings since it allows us to ignore just those types with large numbers of postings. In fact, the resulting computation takes less than $V T^2$ iterations. In practice, we have found that T can often be set quite small. Fig. 5, for example, was computed with $T = 20$, so that the entire calculation took less than $400V \approx 52,000,000$ steps. If we had tried to use the N^2 algorithm, the calculation would have required $37,000,000^2$ steps, which is utterly impractical.

```
for(type=0; type<V; type++) {
  w = weight(type);
  f = freq[type];
  /* the key approximation */
  if(f < T) {
    postings = get_postings(type);
    for(p1=0; p1 < f; p1++) {
      i = postings[p1];
      for(p2=0; p2 < f; p2++) {
        j = postings[p2];
        fimage[CELL(i)][CELL(j)] += w;}}}
```

5.3 Computing the Q-image

After computing the f-image, the floating point values are quantized to conform to the available display hardware. Suppose, for example, that the hardware is designed to handle at most C colors, where $C \approx 256$. An obvious quantization technique is linear interpolation. Unfortunately, we have found that the values in the f-image often belong to an extremely skewed distribution, as shown in Fig. 13. Using linear interpolation on such a highly skewed distribution would introduce serious quantization errors.

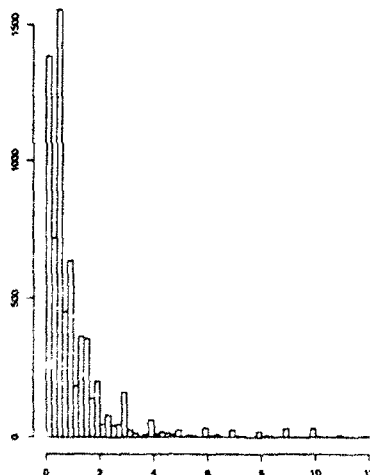


Fig. 13. Histogram of Values in Fig. 8's F-image

We have had more success with a non-parametric approach: histogram equalization (Gonzalez & Wintz 1987, pp. 146-152), which quantizes the values in the f-image into C quantiles, one for each color. Unfortunately, even histogram equalization has difficulties when the input is highly quantized. We have found empirically that many of the f-image values are small integers and ratios of small integers. This might be expected in the text application where Zipf's Law would predict most word frequencies to be small integers; it also appears to hold in the other applications, as well. In order to avoid assigning multiple colors to the same integer, we have found it useful to remove duplicate values before applying histogram equalization.

5.4 User Interface

Finally, the q-image is converted into an image suitable for displaying in a window as a component of the interactive dotplot browser (see Fig. 1). In a color X Windows implementation (Scheifler & Gettys 1986), this final step establishes a mapping from values in the q-image to cells in the X server's default colormap. In a monochrome implementation, the q-image step is unnecessary since the f-image can be converted directly into black and white using various standard techniques such as thresholding, dithering, error diffusion, etc.

In addition to the dotplot views discussed thus far, there are also text views, as shown in Fig. 1c. A text view consists of two panes so that two subsequences of the input can be presented side-by-side. The text view is linked to a dotplot view, so that clicking the mouse on a point in the dotplot corresponding to the pair of tokens x, y causes the left pane to be centered around x and the right pane to be centered around y .

6. Conclusion

Dotplots, which have been used to study homology in biology, are also useful for discovering potentially important patterns in text and source code. In the software application, for example, we have seen that dotplots can be used to discover large-scale structures, remove undesirable duplication when possible, and cope more effectively with duplications than cannot be removed. Similarly, there are also a number of practical ramifications of these patterns in the text application, as well.

We have seen that many of these potentially important patterns are often associated with certain features in the dotplot: diagonals, squares, textures and combinations thereof. There was a considerable discussion of a number of mechanisms that explain some of these associations. Much of the discussion used the browser to analyze a feature in a real sequence, and then tried to replicate the feature in a synthesized dotplot. For example, the browser was used to find

broken diagonals in AP stories (Fig. 4), a combination of squares and diagonals in the Hansards (Fig. 5), and "shrinking diagonals" in a large program (Fig. 9). Each of these features were replicated in a synthesized dotplot: broken diagonals in Fig. 2c, the combination of squares and diagonals in Figs. 6 and 7, and "shrinking diagonals" in Fig. 10. The discussion then concluded with a speculation of the underlying mechanism. In the AP news, for example, we believe the diagonals were probably broken by the insertion of a few extra facts into a rewrite. Similarly, we believe the shrinking diagonals in the software example were probably caused by a repeating sequence of initializations to 0 being diluted with increasing numbers of non-zero initializations.

In many cases, the patterns are much easier to find with a dotplot than with an alternative such as a text editor or the UNIX diff program. A text editor, for example, is ill-suited for identifying structures that extend well beyond the size of the screen. Similarly, the diff program is ill-suited for identifying a texture such as the "shrinking diagonals" pattern discussed in Section 4.2, because the diff program attempts to find a single alignment path and therefore can't deal effectively with the rich structure of multiple overlapping matches.

The final section of the paper described the implementation of dotplots, with an emphasis on weighting, compression and approximation. These steps make it possible compute dotplots quickly enough for use in an interactive browser.

Acknowledgements

We would like to thank Joe Kruskal for pointing us to the biology literature, and Justina Voulgaris for providing us with the pBR322 and pUC18 data as well as pointers to the appropriate references. The X implementation benefited considerably from Doug Blewett's invaluable knowledge. We also appreciate the enthusiastic response from early users of the browser, especially Chris Ramming. Eric Sumner and Joe Steffen provided access to the SESS code and an appreciation of the realities of programming in the large.

REFERENCES

- Argos, P. 1987. A Sensitive Procedure to Compare Amino Acid Sequences. *Journal of Molecular Biology* 193:385-396.
- Baker, B. S. March 1992. A Program for Identifying Duplicated Code. *INTERFACE '92*. Interface Foundation of North America.
- Balbas, P.; X. Soberon; E. Merino; M. Zurita; H. Lomeli; F. Valle; N. Flores; and F. Bolivar. 1986. Plasmid Vector pBR322 and its Special-Purpose Derivatives — A Review. *Gene* 50:3-40.
- Blundell, T. L.; B. L. Sibanda; M. J. E. Sternberg; and J. M. Thornton. March, 1987. Knowledge-Based Prediction of Protein Structures and the Design of Novel Molecules. *Nature* 326:263-272.
- Carrington, J. C., and T. J. Morris. 1987. Structure and Assembly of Turnip Crinkle Virus IV. Analysis of the Coat Protein Gene and Implications of the Subunit Primary Structure. *Journal of Molecular Biology* 194:265-276.
- Doolittle, R. F. October 1961. Similar Amino Acid Sequences: Chance or Common Ancestry?. *Science* 214:9:149-159.
- Eick, S. G. March 1992. *Dynamic Graphics for Software Visualization*. INTERFACE '92. Interface Foundation of North America.
- Gonzalez, R. C., and P. Wintz. 1987. *Digital Image Processing*. Addison-Wesley. Second Edition.
- Lake, J. A.; V. F. de la Cruz; P. C. G. Ferreira; C. Morel; and L. Simpson. July 1988. Evolution of Parasitism: Kinetoplastid Protozoan History Reconstructed from Mitochondrial rRNA Gene Sequences. *Proceedings of the National Academy of Science, USA* 85:4779-4783.
- Laver, W. G.; G. M. Air; T. A. Dopheide; and C. W. Ward. January 1980. Amino Acid Sequence Changes in the Haemagglutinin of A/Hong Kong (H3N2) Influenza Virus During the Period 1968-77. *Nature* 283:31:454-457.
- Maizel, J. V., and R. P. Lenk. December 1981. Enhanced Graphic Matrix Analysis of Nucleic Acid and Protein Sequences. *Proceedings of the National Academy of Science, USA* 78:12:7665-7669. *Genetics*.
- Pustell, J., and Fotis C. Kafatos. 1982. A High Speed, High Capacity Homology Matrix: Zooming Through SV40 and Polyoma. *Nucleic Acids Research* 10:15:4765-4782.
- Quax-Jeuken, Y. E. F. M.; W. J. Quax; and H. Bloemendal. June 1983. Primary and Secondary Structure of Hamster Vimentin Predicted from Nucleotide Sequence. *Proceedings of the National Academy of Science, USA* 80:3548-3552. *Biochemistry*.
- Salton, G. 1989. *Automatic Text Processing*. Addison-Wesley.
- Scheifler, R. W., and J. Gettys. April, 1986. The X Window System. *ACM Transactions on Graphics* 5:2:79-109.
- Tufte, E. R. 1983. *The Visual Display of Quantitative Information*. Graphics Press.
- Tukey, J. W. 1977. *Exploratory Data Analysis*. Addison-Wesley.
- Vingron, M. 1991. *Multiple Sequence Alignment and Applications in Molecular Biology*. Heidelberg University dissertation.
- Yanisch-Perron, C.; J. Vierira; and J. Messing. 1985. Improved M13 Phage Cloning Vectors and Host Strains: Nucleotide Sequences of the M13mp18 and pUC19 Vectors. *Gene* 33:103-119.

Dynamic Graphics for Software Visualization

Stephen G. Eick
AT&T Bell Laboratories
Room IHC 1G-339
1000 E. Warrenville Road
Naperville, Illinois 60566 U.S.A.

19 February 1991

Keywords: software visualization, dynamic graphics, code browsing, change management systems

SUMMARY

One of the biggest challenges in computing is understanding the source code in large software systems. The source code listing for even a moderate sized system of 10,000 lines runs to hundreds of pages. The volume of code makes it difficult for programmers to understand the underlying structure. For programmers to make even small changes may require several weeks of detailed study. To help understand code I have developed a software tool, *Seesoft*TM, that applies dynamic graphics techniques to the problem of visualizing software.

Seesoft displays a directory of source code by showing each file in the directory as a rectangle whose height corresponds to the size of the file. Each line in the file is shown as a row within the rectangle whose length and indentation correspond to the actual code. The lines are colored (running through a spectrum from red to blue) according to a statistic obtained from the version control history such as age, programmer, or feature. The visual impression is that of a miniature picture of all the source code with the indentation showing the control structure and the color showing the age.

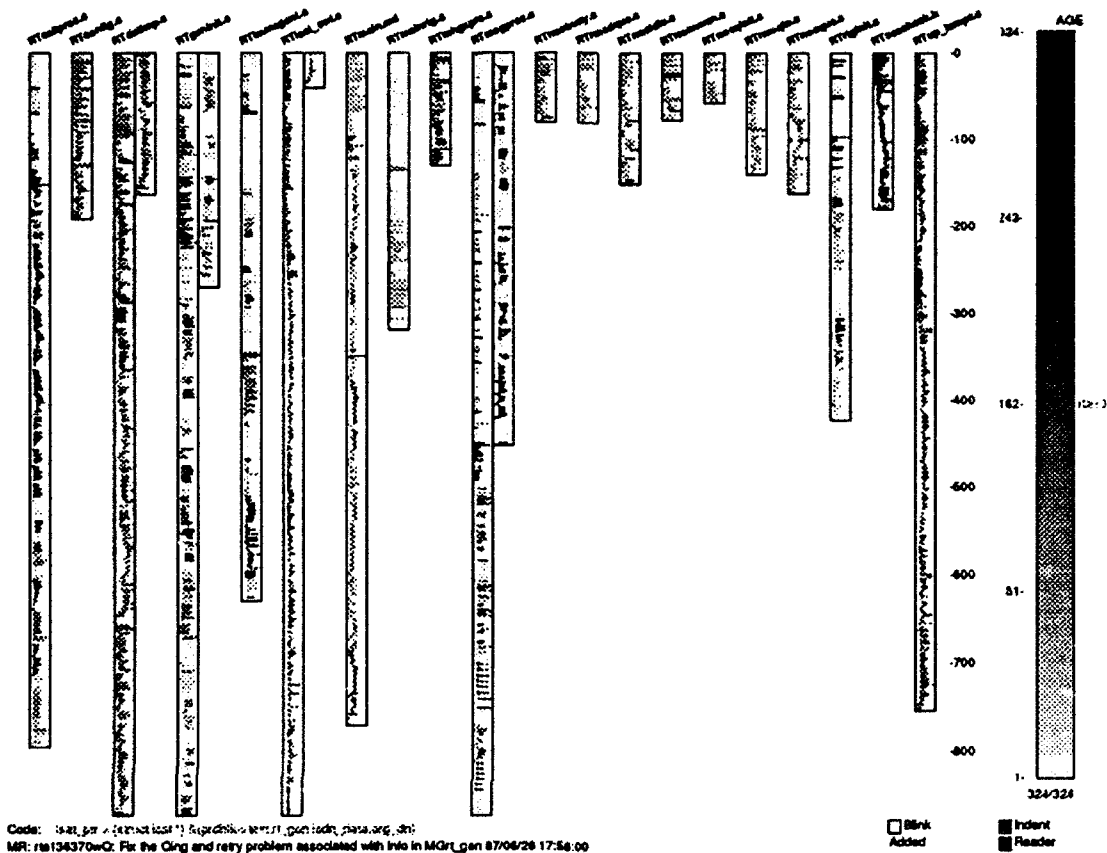
A key idea in *Seesoft* is to enable the user to manipulate the display to gain insight into the code. *Seesoft* uses high-interaction, dynamic graphics techniques to enable the user to interact with the code and discover previously unknown characteristics. Using *Seesoft* it is possible to view up to 50,000 lines of code on a high-resolution 19 inch color monitor.

1. INTRODUCTION

One of the biggest challenges in computing is that of software productivity. Nowhere is this more important than in large multi-programmer, multi-year projects, where hundreds or thousands of programmers work together on huge systems. A big problem in these systems is understanding the source code, its history, and how it all fits together. This is particularly difficult when the programmers who maintain the code are different from those who wrote it, perhaps due to staff turnover or new project assignments. Programmers, given requests for additional functionality, must study the current code to determine which files contain the existing functionality and which lines to change within these files. This task is often difficult and time consuming. In fact, it may take several weeks of detailed study to change a few lines with no

unwanted side effects. Votta (1992) estimates that on large old projects as much as 90% of a new programmer's time and 70% of an experienced programmer's is used to study code before changing it. The studying is often done by looking at code listings and by viewing the code with editors on terminals.

The source code listing for even a moderately sized system can be huge. The listing for a system containing 10,000 to 100,000 lines, printed 50 lines per page, would run 200 to 2,000 pages and the listing for a large system with 1 million lines would require 20,000 pages. Because of the volume of code it is difficult for programmers to gain insight into the code's structure using editors that can display at most a couple hundred lines or looking at code listings. Better methods are needed.



I decided to study the problem of understanding source code from a data analysis perspective. My motivation comes from studying code for a large real-time telephone switch. The source code for this system, as well as all production software systems, is placed in a change management system. The common change management systems widely used include the Revision Control System (Tichy 1985), Source Code Control System (Rochkind 1975), Change Management System (Rowland and Welsch 1983), Extended Change Management System (Tuscany 1987), and SABLE (Cichinski and Fowler 1988). These systems maintain a complete change history of the source code and can recreate it as it existed at any point in time. The change management system used by the switching system also stores related variables to help manage the project such as the

date, reason for the change, responsible programmer, affected feature, whether the change fixes a bug or adds a new feature, etc. The change history is a rich, underutilized, resource of information about the system.

To analyze this class of data I apply dynamic graphics methods (Becker and Cleveland 1987) and have invented a new software tool, Seesoft™, (Eick, Steffen, and Sumner 1992), embodying the method. I call the emerging field devoted to visually displaying software *Software Visualization*. The visualization technique is to represent each file as a column whose height corresponds to the size of the file. Within each column rows represent each line in the file. The length and indentation of each row corresponds to that of the actual line. Each row is colored according to a statistic associated with that line

such as the age of the line, programmer who created it, the feature that it is for, or the type of line. The statistic may be obtained from the change management system or from the code itself.

Figure 1¹ shows a sample Seesoft display for a directory containing 20 source code files with 9,365 total lines of code, colored according to age. From the figure the first observations concern the sizes of the files, locations of the control structures, and age of the code. The length of each column tells how large each file is. Files longer than one column are continued over to the next column. The indentation and length of each line looks the same as in the source file, thus showing information about C language (Kernighan and Ritchie 1988) control structures. The color of each line² shows its age. The newest lines are in red (black) and the oldest in blue (light gray). On the right there is a scale showing the color for each change to this directory. In total there have been 324 changes. The visual impression is that of a miniature picture of all the source code with the indentation showing the usual C control structure and the color showing the age.

A key idea in Seesoft is direct display manipulation (Shneiderman 1983). Seesoft enables the user to manipulate the display to gain insight into the code. As will be described in more detail below, using the mouse a programmer may activate or deactivate individual changes, lines, files, date ranges, or even types of changes. To view the code text a programmer may open code reader windows. Seesoft increases the effectiveness of these operations by using techniques from high-interaction graphics in which the manipulations are performed in real-time.

The remainder of this paper describes the techniques in more detail. Section 2 describes the Seesoft tool and its application of dynamic graphics methods. Section 3 discusses its use for analyzing software change history. Section 4 tells how I implemented Seesoft. And finally, Section 5 concludes.

2. THE SEESOFT TOOL FOR SOFTWARE VISUALIZATION

An important idea in dynamic graphics is to display statistical data graphically and then to manipulate the display, parameters describing the display, or animate, to gain insight into the data. Becker, Eick, and Wilks (1990) call this "parameter focusing." To apply this technique to software change history it is necessary to represent software graphically. This is particularly difficult for software because of the large volume of source code.

2.1 Graphical Data Representation and Screen Layout

As shown in Figure 1, Seesoft displays source code by representing each file as a rectangle and every line in the file as a row within the rectangle. This display is similar to that of Baeker and Markus (1990, p. 235) who show a reduced representation of code that has been typeset. This compact representation can comfortably display 20 files, 1,000 lines each, on a standard high resolution workstation color monitor and has displayed 50 files. The file names are shown on an angle above each rectangle. The indentation and length of the rows corresponds to lines of code. The representation is reduced as much as possible so that the C control structures, *case*, *if*, *and*, *function calls*, are clearly visible on a high resolution monitor, although they may be difficult to see when reduced to fit in this paper.

The color of each row is determined by a categorical statistic associated with the line. In Figure 1 the statistic is the age. On the right-hand side of the display there is a mouse sensitive color scale coded with a discrete color for each statistic value. At the bottom of the color scale the number of active and total number of statistic values is shown, 324 out of 324, in Figure 1.

At the bottom of the screen are buttons and toggles that control the display mode. To the left of the buttons there are two rows for displaying text. On these two lines I print an abstract for the change associated with the current mouse position and the actual line of code if the mouse is inside one of the rectangles.

For this display approach to be effective the initial display must be clear and informative. With the Seesoft display programmers immediately recognize the files and lines of code because Seesoft looks like a code listing viewed from a distance. The initial view shows the relative sizes

1. The display techniques are optimized for color monitors.

2. In the black and white figures in this paper "color" is to be interpreted as gray level.

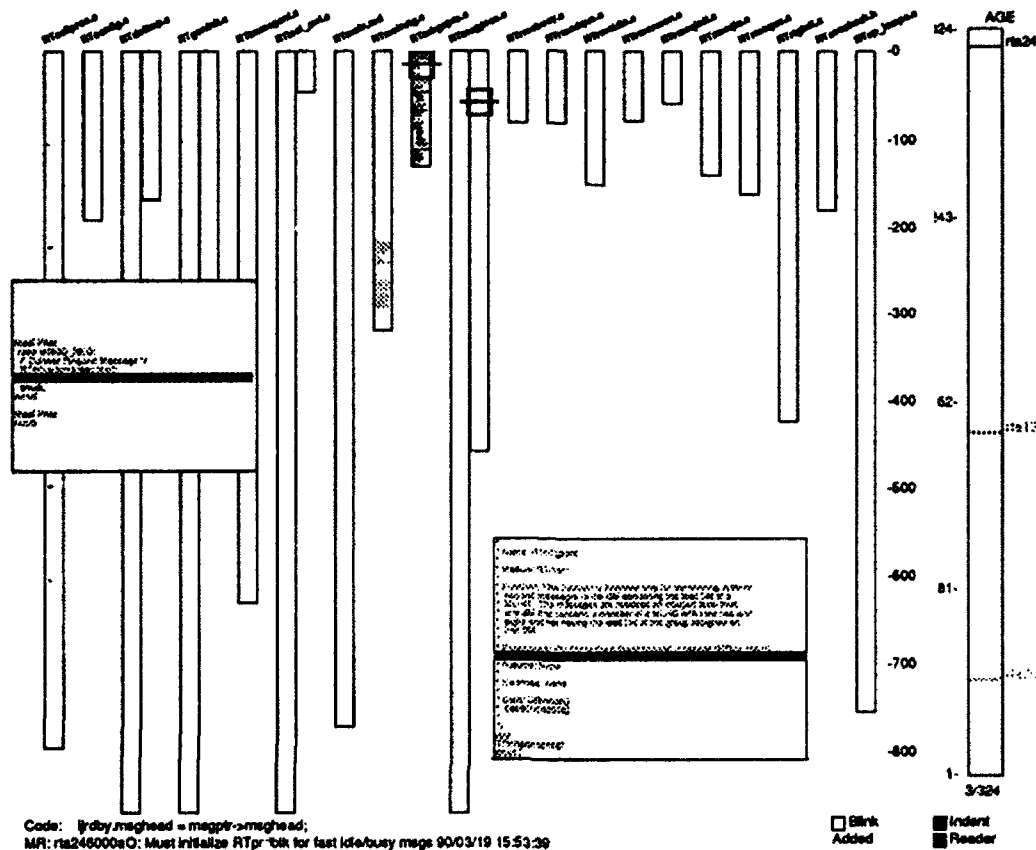


Figure 2. Sequence Of Changes

As the user positions the mouse over a change, the lines of code that change created are activated. Two *Reader* windows are shown. The code underneath the magnifying boxes appears in the windows.

of the files, the longest files span multiple columns. The spatial distribution of the colors shows how the statistic is distributed in the code.

There must be easy and intuitive human interface techniques for the user to manipulate the display. I find that using direct manipulation techniques from dynamic graphics, in particular "brushing," and "linking," allow the programmers to find interesting patterns in the data. Programmers need to be able to see the actual source code and not just a representation of it. For this purpose Seesoft users may open up *Reader* windows to display the text underneath "magnifying" boxes that track mouse movement (shown in Figure 2). This technique is particularly effective because the programmer has both an overview of the code and also can see parts of the code in detail.

2.2 Brushing

Seesoft uses the brushing technique invented by Becker and Cleveland (1987). The color scale, file names, and lines on the Seesoft screen are mouse sensitive. As the mouse is moved around the screen the object currently under the mouse is activated and deactivated after the mouse moves away. The left mouse button causes the activation to be permanent and the middle mouse button deactivates previously activated entities. For different entities activation has different meanings.

Activating the color corresponding to a date turns on its color and also turns on any code associated with that date. In Figure 1 all dates have been activated and correspondingly all lines of code are visible. Activating a line of code turns on its color, the color of its corresponding date, and any other lines associated with this date. Activating a

file activates all lines in the file and so on. When the mouse is over a date, the abstract for that change made on that day is displayed on bottom display line and when the mouse is over a line of code both the code and abstract are displayed on the bottom two lines. Figure 2 shows an example where three dates have been activated and there are two reader windows.

Brushing is most effective when the manipulations are performed in real-time. It is difficult to illustrate this technique in a static medium such as this paper. But the brushing technique is particularly good for scanning through lots of dates and changes.

2.3 Code Reading

Programmers, upon discovering interesting patterns, need to see the actual code. When the programmer depresses the *Reader* window button two actions occur. A new window for displaying code using a 10 point font is created and small colored "magnifying" box appears on the graphical display. As the magnifying box is moved around the display code underneath it is displayed in the window. The size of the magnifying box is proportional to the size of the reader window. This enables the programmer to understand what fraction of the total is visible, where the code is in the file, and which file in the directory.

Multiple reader windows may be created and independently positioned. The border on each reader window has a unique color that is tied to its corresponding magnifying box. The programmer manipulates and positions the boxes independently using the mouse and right mouse button. Depressing the right mouse button positions the active reader window and depressing the right mouse button again grabs the closest reader window if all are positioned.

2.4 Linking

Linking between displays is a powerful technique for understanding data (Stutzle, 1987) and (Haslett, Bradley, Craig, Unwin, and Wills 1991). In Seesoft the code reader windows are linked to the base display in two ways. First, as the magnifying box is moved around the display the code in the window changes. This enables the programmer to have a "birds eye" view of all the code and yet see the particular lines of interest. Second, the colors of the text lines in the reader windows are linked to the colors of their line representations. As the programmer manipulates

the Seesoft display to select and deselect entities thereby changing their color, the color of the code in the reader windows is continuously updated to reflect the current display state.

The lines of code and changes are also linked. As the mouse touches a change, the affected lines are activated. Similarly, when the mouse touches a line, the change associated with that line is activated along with all other lines associated with that change.

2.5 Display Modes

In the lower right-hand corner of the Seesoft display are buttons and toggles that modify the current display mode. By default, Seesoft's line representation tracks the indentation of the text lines. The programmer may turn off indentation using the *Indent* button. This is useful for displaying lots of code when the files are close together.

It is possible to display two statistics for each line using the "split column" feature. Programmers might use the feature to display the dates that lines were added and deleted. The *Added/Deleted/Both* toggle has three modes. *Added* associates each line with the first statistic, *Deleted* with the second, and *Both* causes each column to be split down the middle with the first statistic on the left and second on the right.

The *Blink* button is used to study the spatial distribution of another binary statistic. When activated, lines associated with this binary statistic blink.

3. APPLICATIONS IN SOFTWARE ENGINEERING

The motivation for this research comes from studying the source code and software structure in a very large real-time switching system. The analysis techniques embodied in Seesoft have application in several areas of software engineering including, code discovery, project management, new programmer training, and code archaeology studies. To illustrate the use of Seesoft I will analyze data from a sample directory. This directory includes about 20 C source code files ending in .c, C header include files ending in .h, and a configuration file ending in .md.

Figure 1 shows all of the files in this directory with the color of each line tied to its age. There have been 324 changes to the code in this

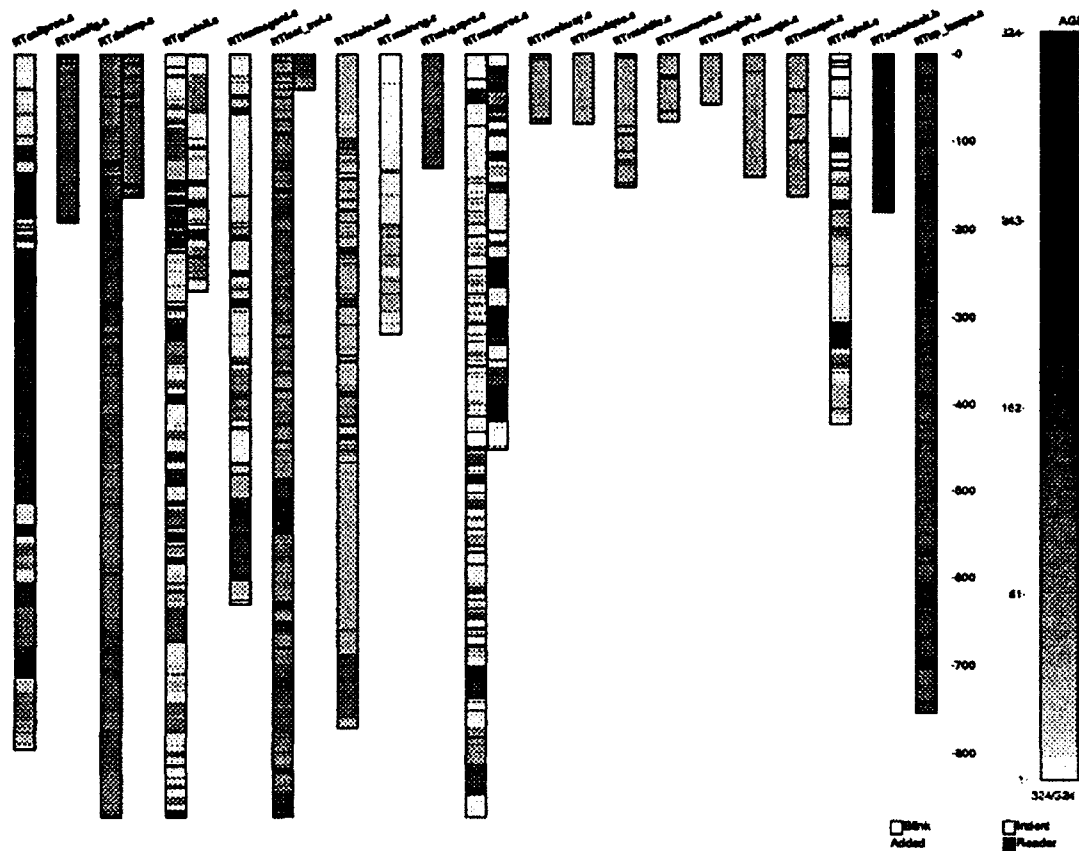


Figure 3. Source Code With No Indentation

By turning off indentation it is easier to see the age of the code. The oldest lines are displayed in dark blue and the newest in red. The display shows the relative size of the files, age of the code, and how many times each file has been changed.

directory since 1984. From the length of the columns, the largest file, RTmsgproc.c, has about 1,500 lines and the smallest, RTmsginit.c, has about 50. The indenting patterns show the locations of the C control structures. The sawtooth blocks in RTup_lamps.c are the *cases* in a large C *switch* statement. The scalloping are indentation for the C *if* statements and C *for* loops.

Figure 3 is the same as Figure 1 except the code indenting has been turned off. Turning off indenting makes it easier to see the age of the code. The most recently added lines are shown in red (black)³ and the oldest in blue (light) according to a rainbow scale (Levkowitz and

Herman 1992). The age patterns in the files are striking. Much of the code in files RTmsgproc.c, RTcallproc.c, RTgeninit.c, RTmainrto.c and RTginit.c is blue indicating that it dates to 1984-1985. It is difficult to relate the colors from the spectrum to calendar dates, but the exact date and time code was added is encoded in the change abstract that is printed below when changes are activated. These files are interesting because along with the blue code they display many other colors indicating that they have been changed many times, including recently⁴. There is another

2. The indenting is difficult to see in the figures, but shows clearly on standard 19 inch high-resolution color monitors.

3. In the black and white rendering of this display I use a gray scale instead of a color spectrum. My printer unfortunately uses half-tones to produce the gray scale that makes it difficult to read.

4. An expert on this software explained that these files are the main control for this process.

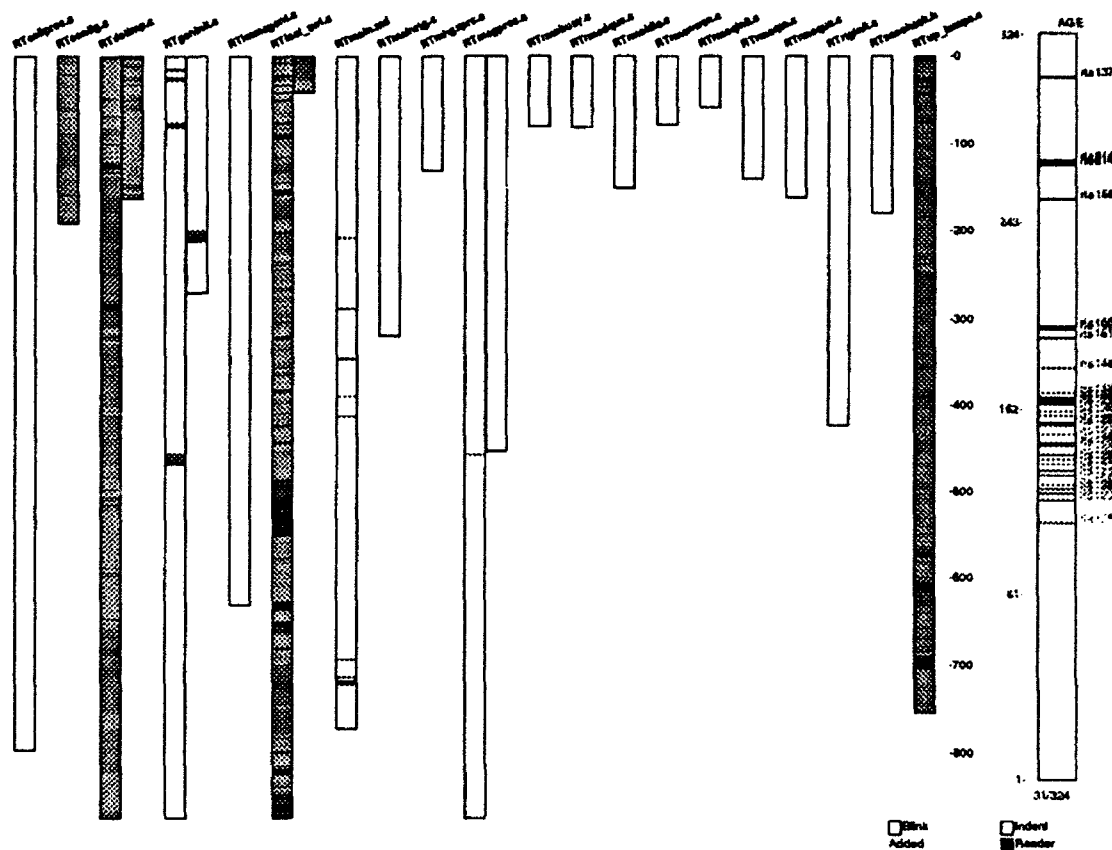


Figure 4. A Significant New Feature

A new feature was put into the code in 1987. This code is primarily in 4 files and has been stable since its introduction.

set of small yellow files dating to 1985, and a set of green files dating to 1987. The yellow and green files have been relatively stable since they were created with a few recent changes.

A good strategy for enhancing software is to add a few lines to the main part of a system and put the new code in its own file. Figure 2 shows this type of modular enhancement. File RTmhgxprc.c was created with a single change (the middle change on the color spectrum in Figure 2) and is a C function that is shown in the first reader window. The code to execute this function was added in the same change to RTmsgproc.c and is shown in the second reader window.

Files that have been frequently changed are often difficult to maintain and are candidates to be rewritten. Although not illustrated in a figure, by deactivating all changes and sequentially brushing each of the file names, RTmsgproc.c is the most frequently modified and has been changed 65

times. This file is called a "rainbow" file after the spectrum of colors it contains.

By activating files created on common dates, I discover that there are three different sets of files in this directory. Each set was created by the same class of changes. Figure 4 shows one of the sets, a major enhancement done in 1987. The functionality in these four files is self contained except for a few function calls in the other files.

Which files have bugs? Figure 5 shows the locations of the bugs by deactivating code added to fix them. It is interesting in that comparing Figures 3 and 5, the bug fixes are more concentrated in the heavily changed files. These files might be candidates to be rewritten. By using the split-column mode, it is possible to show the bug fixes that were added to fix previous bug fixes or "fix-on-fixes."

From this data analysis session what have I learned? I know the sizes of the files, which are

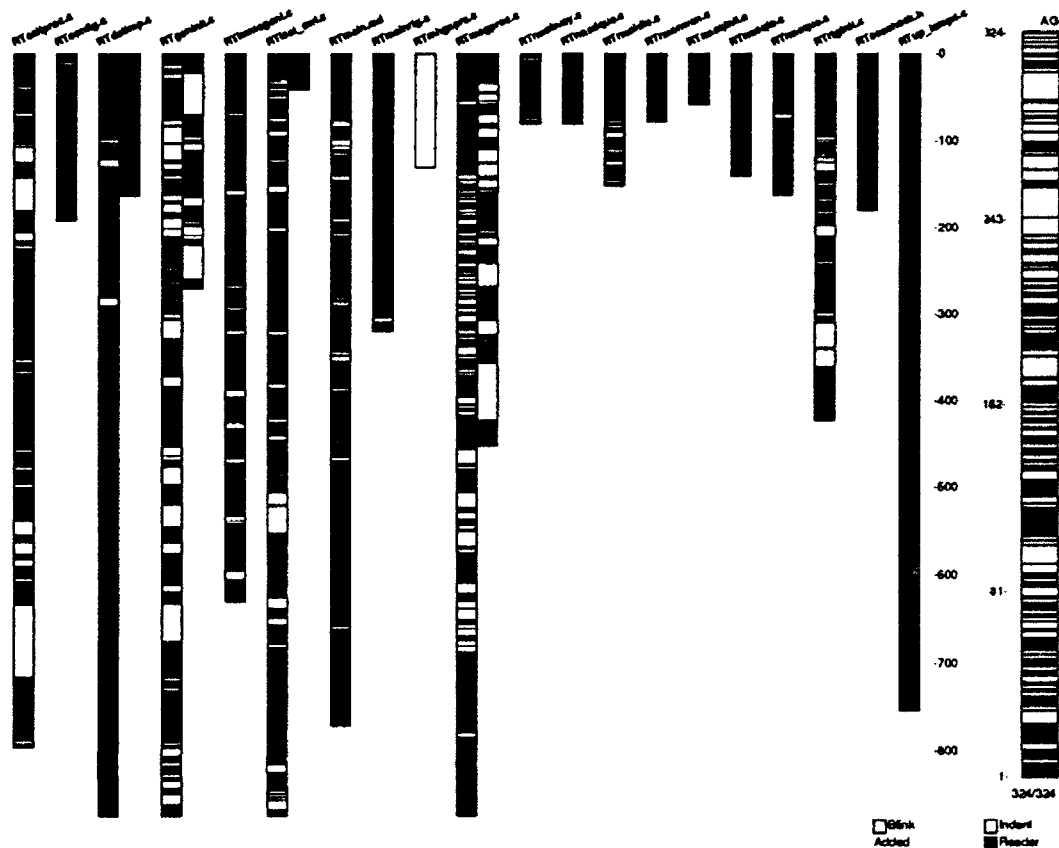


Figure 5. Locations of Bug Fixes
Changes to fix bugs have been deactivated. The bug fixes are concentrated in a few of the files.

stable and which have been changed. I have discovered several modular enhancements and also a major feature enhancement. I know which code has been recently added, which code is old, and what files have been recently changed. I have discovered that there are three distinct classes of files in this directory, each class created at different times with different maintenance characteristics. If this directory ever got too big, these would be natural candidates for splitting. I also found that bug fixes are more concentrated in certain files that have been frequently changed.

4. IMPLEMENTATION

Seesoft is written in C++ (Stroustrup 1987), and currently runs on Silicon Graphics Iris workstations and other workstations supporting GL (Silicon Graphics 1990) graphics library such as IBM's RS6000. Eventually, I plan to port Seesoft to the X Window System (Quercia and O'Reilly 1987). The total code for Seesoft is

about 2,500 lines. Seesoft uses C++'s object oriented capabilities (Coplien 1992) to simplify the coding. All entities on the screen are represented as C++ classes and inherited from a base entity class. The redrawing and mouse manipulation is accomplished by executing the virtual member function for the appropriate entity on the screen.

For high graphics performance much of the user interface is done using color map manipulation (Foley, van Dam, Feiner, Hughes 1990). Seesoft draws each line and associated data with its own color. Then activating, deactivating, and linking is done by manipulating the color map. The colors for the activated MRs are turned on and for the deactivated MRs are turned off. Color map manipulation is fast because it is done in hardware. Standard color maps often have 256 different colors and so to display more than 256 changes requires a large color map.

Silicon Graphics Iris workstations come with 19 inch color monitors with 1280x1024 screen resolution. Using the columns and rows representation of files and lines I can easily display 20 columns and have displayed 50 columns on a single monitor. With more than 50 columns the lines displayed by the row representation become too thin. Each column may represent 1,000 lines.

For static graphics and preliminary analysis I use the S language (Becker, Chambers, Wilks 1988). S provides a computational environment, static graphics, and data management that support the interactive manipulations. I link Seesoft into the S executive, perform all data manipulation in S, and then launch Seesoft from S.

The figures in this paper were created by including PostScript (Adobe Systems, 1987) files into troff input. The technique Seesoft uses to produce PostScript images of the current display is due to Wilks. When Seesoft's PostScript option is selected, Seesoft redraws the display and simultaneously writes color PostScript code into a file for each GL graphics call. Then, by prepending a PostScript preamble, I create an encapsulated PostScript file that is an exact replica of the current Seesoft display for printing. I produce black and white PostScript files by manipulating the color map.

5. CONCLUSIONS

This paper describes a new dynamic graphics technique for visualizing the source code and a tool, Seesoft, embodying the technique. I call this new area of trying to visually display source code *Software Visualization*. Seesoft develops a graphical representation for code and applies dynamic graphics techniques to manipulate the representation.

Seesoft represents files as rectangles and the lines in the files as rows within the rectangles colored according to a statistic associated with each line. Seesoft allows the user to manipulate the display in real-time using high-interaction graphical techniques. The display is mouse sensitive and, as the user moves the mouse around the screen, entities are activated and deactivated. The entities are linked together through common statistic values and color. The users manipulate the display with mouse movement and buttons to discover interesting patterns. Seesoft presents the user with multiple views of the code, one of which may be reader window that displays the actual

code text (in the active colors). The visual representation of code allows a programmer to gain the insight and understanding that goes with visualization and the reader window allows the user to see the actual code.

Besides software, the display and manipulation techniques have application to other ordered databases. The approach is applicable to databases where there is interest in understanding the overall structure and querying the database based on particular attributes uniquely associated with each entity. For example, I could display a text corpus such as the Bible. Each book could be represented as a column and each verse as a row. I could order the database using a subject index. Other applications include legal writings and software documentation. Another possible application would be to use each column in Seesoft as a sophisticated scroll bar in a text editor (Hill, Hollan, Wroblewski, McCandless, 1991).

My motivation for developing these techniques comes from studying the change history in a large software system. The change management systems allow me to recreate the code as it existed at any point in time. With Seesoft I can comfortably display 20,000 lines of code and have displayed 50,000 lines. By displaying this volume of code I obtain insights and a perspective on the system and its evolution that would otherwise be impossible.

Acknowledgements

I would like to gratefully acknowledge helpful conversations with Richard A. Becker, Allan R. Wilks, and Eric E. Sumner, PostScript help from Rich Drechsler, and database help from Joseph L. Steffen.

REFERENCES

1. Adobe Systems Incorporated (1990). *PostScript Language Reference Manual*, Second Edition, Addison-Wesley, Reading, Massachusetts.
2. Baeker, R., and Marcus, A., (1990). *Human Factors and Typography for More Readable Programs*, Addison-Wesley, Reading, Massachusetts.
3. Becker, R.A., Cleveland, W. S., (1987). "Brushing scatterplots," *Technometrics*, Vol. 29, 127-142.
4. Becker, Richard A., Cleveland, William S., Wilks, Allan R. (1987). "Dynamic graphic for data analysis," *Statistic Science*, Vol. 2,

- 355-395.
5. Richard A. Becker, John M. Chambers, and Allan R. Wilks (1988), *The New S Language*, Wadsworth & Brooks/Cole, Pacific Grove, California.
 6. Becker, R. A., Cleveland, W. S., and Weil, G. (1988). "The Use of Brushing and Rotation for Data Analysis," *Dynamic Graphics for Statistics*, William S. Cleveland and McGill (Eds.), 247-275, Wadsworth & Brooks/Cole, Pacific Grove, CA.
 7. Becker, R. A., Eick, S. G., Miller, E. O., Wilks, A. R. (1990). "Dynamic Graphical Analysis of Network Data," *Interface '90 Proceedings*, East Lansing, MI.
 8. Cichinski, S. and Fowler, G. S. (1988). "Product Administration Through SABLE and NMAKE," *AT&T Technical Journal*, Vol. 67(4), 59-70.
 9. Coplien, J. O. (1992). *Advanced C++ programming styles and idioms*, Addison-Wesley, Reading Massachusetts.
 10. Eick, S. G., Steffen J. L., and Sumner, E. E. (1992). *Seesoft—A Tool For Visualizing Software*, Submitted to *IEEE Transactions On Software Engineering*.
 11. Foley, J. D., van Dam, A., Feiner, S. K., Hughes, J. F. (1990). *Computer Graphics Principles And Practice*, Second Edition, Addison-Wesley, Reading Massachusetts.
 12. Haslett, J., Bradley, R., Craig, P., Unwin, A., and Wills, G. (1991). "Dynamic Graphics for Exploring Spatial Data With Application to Locating Global and Local Anomalies", *The American Statistician*, Vol. 45., No. 3.
 13. Hill, W. C., Hollan, J. D., Wroblewski, J., McCandless, T. (1991). "Edit wear and read wear: their theory and generalizations", CHI-91.
 14. Kamada, T. (1989). *Visualizing Abstract Objects and Relations*, World Scientific, Teaneck, New Jersey.
 15. Kernighan, B. W. and Ritchie, M. (1988). *The C programming language*, Prentice-Hall, Englewood Cliffs, New Jersey 07632.
 16. Levkowitz, H. and Herman, G. T. (1992). "Color Scales for Image Data", *IEEE Computer Graphics and Applications*, Vol. 12, No. 1.
 17. Quercia, V. and O'Reilly, T. (1987). *X Window System User's Guide*, O'Reilly & Associates, Inc., Sebastopol, CA.
 18. Rochkind, M. J. (1975). "The Source Code Control System," *IEEE Transactions on Software Engineering*, Vol. SE-1(4), 364-370.
 19. Rowland B. R. and Welsch, R. J. (1983). 'Software Development System,' *Bell System Technical Journal*, Vol. 62(1) Part 2, 275-289.
 20. Shneiderman, B. (1983). "Direct Manipulation: A Step Beyond Programming Languages," *IEEE Computer*, Vol. 16(8), 57-68.
 21. Silicon Graphics (1990). "Graphics Library Reference Manual", Document 007-1203-040.
 22. Stroustrup, B. (1987). *The C++ Programming Language*, Addison-Wesley, Reading MA.
 23. Stutzle, W. (1987). "Plot windows," *Journal of the American Statistical Association*, Vol. 82, 466-475.
 24. Tichy, W. F. (1985). "RCS—A System for Version Control," *Software—Practice and Experience*, Vol. 15(7), 637-654.
 25. Tuscany, P. A. (1987). 'Software Development Environment for Large Switching Projects,' *Proceedings of International Switching Symposium*, Phoenix, Arizona, 199-214.
 26. Voua, L. (1992). Personal Conversation.

An Approach to Statistical Spatial-Temporal Modeling of Meteorological Fields

Mark S. Handcock

Department of Statistics & Operations Research
Leonard N. Stern School of Business
New York University
New York, NY 10006

James R. Wallis

Department of Mathematical Sciences
IBM T. J. Watson Research Center
Yorktown Heights, NY 10598

Abstract

In this paper we develop a random field model for the mean temperature over the region in the northern United States covering eastern Montana through the Dakotas and northern Nebraska up to the Canadian border. The readings are the temperatures at the stations in the United States historical climatological network reported in Quinlan, Karl & Williams (1987).

The stochastic structure is modeled by a stationary spatial-temporal Gaussian random field. For this region, we find little evidence of temporal dependence while the spatial structure is temporally stable. The approach strives to incorporate the uncertainty in estimating the covariance structure into the predictive distributions and the final inference.

As an application of the model we derive posterior distributions of the areal mean over time. A posterior distribution for the static areal mean is presented as a basis for calibrating temperature shifts by the historical record. For this region and season, it indicates that under the scenario of a gradual increase of $5^{\circ}F$ over 50 years, it will take 20-30 years of data before the change will be discernible from the natural variation in temperatures.

Key words: Gaussian random fields; Bayesian statistics; Climatic change.

1. INTRODUCTION

There has been much interest recently in climatic change and potential global warming. Of central focus is the phenomenon popularly called the "greenhouse effect": the heating of the earth via the entrapment, by certain gases, of long-wave radiation emitted from the earth's surface. This effect produces a global mean temperature of about $59^{\circ}F$ rather than an estimated $-6^{\circ}F$ in the absence of atmosphere (Mitchell (1989)). Increasing concentrations of the gases thought to contribute to

this effect have led to concern in the scientific community about increases in temperature and the resulting climatic effects.

There appears to be no clear cut consensus on the extent of global warming over the last century. Most estimates run from $0.5^{\circ}F$ to $1.0^{\circ}F$. The difficulty is the lack of good long-term data over large regions. The global temperature constantly changes on time scales of tens of thousands of years. In fact there have been times in the past millennium when it has been much warmer than the majority of global warming scenarios. The question here is a rapid change over the next century that will have enormous impact on the environment.

Much of the evidence for a global warming effect has been based on large-scale Global Circulation Models (GCMs). These use multi-level mathematical representations of the atmosphere for weather prediction. Given the complexity of the environment and the relative simplicity of the models there is much controversy concerning their validity. Results from the four most widely cited GCMs (1) the National Center for Atmospheric Research (NCAR), (2) Geophysical Fluid Dynamics Laboratory (GFDL) of the National Oceanographic and Atmospheric Administration, (3) the Goddard Institute of Space Studies (GISS) and, (4) the Hadley Center for Climate Prediction and Research at Bracknell, England, are still far from being in agreement, although all models predict higher winter temperatures at the higher northern altitudes as a function of increasing greenhouse gases.

Significant global warming would have an enormous effect on the environment and the world economy. Altering the economies of the world to reduce the production of the gases suspected of increasing the greenhouse effect would be very costly and/or drastically alter our way of life. If the political decision is to be postponed until the empirical evidence is in, this paper sheds light on how long must we wait to detect a global warming with high confidence.

In this paper we develop spatial-temporal models for

temperature fields over a region in the northern United States covering eastern Montana through the Dakotas ($90^\circ - 107^\circ$ in longitude) and northern Nebraska up to the Canadian border ($41^\circ - 49^\circ$ in latitude). We choose the winter months and this region as our study area because GCM predictions of climatic change ($4^\circ F - 10^\circ F$) induced by increased greenhouse gases are expected to be at maximum for high latitudes during the winter months (Mitchell (1989), IPCC (1990)). The observed lack of temporal trend for this region and period was somewhat of a surprise. However, this finding has been confirmed using different statistical approaches (Lettenmaier, Wood and Wallis (1992)). It is also evident from this later study that, had other regions or periods been picked, the results would have been quite different. In addition the relatively stable and simple topography of the region help to ensure homogeneity and the minimization of localized effects. Data from the United States historical climatological network reported in Quinlan, Karl & Williams (1987) is being used to explore long term changes and potential effects of increased concentrations of the greenhouse gases.

There is much interest in empirical studies of climatic change. Jones, *et al* (1986) considers station data to investigate long-term variation in the surface temperature of the northern hemisphere. Karl (1984, 1985) considers climate variation and change in North America. These studies emphasize the dynamic nature of the climate system, and the existence of abnormal winter temperatures within the climate system. Other empirical work is reported in Diaz & Quayle (1978, 1980). A hindrance to these and earlier studies has been the dearth of quality data with both spatial and temporal extent.

Karl, Heim and Quayle (1991) consider a similar region with the objective of identifying greenhouse effects. They construct a pure time-series model for the averages of the stations enclosed in the region and do not address the spatial aspects of the temperature field. This paper develops a comprehensive model for the spatial dimension in conjunction with the temporal component. In addition the model is for the meteorological field as a whole, rather than just a particular characteristic. This is important as it facilitates direct comparison with GCMs and prediction of derived quantities throughout the region and over time. In particular, it allows the prediction of the meteorological field at each location (e.g. city or county) with an associated assessment of the quality of prediction.

The traditional best linear unbiased prediction procedure ("Kriging") is used in this paper for inference, but within a Bayesian framework. Particular attention is paid to the treatment of parameters in the covariance

structure and their effect on the quality, both real and perceived, of the prediction.

Our approach is to use posterior distributions for the static areal quantities as a basis for calibrating temperature shifts by the historical record. In particular, the objective is to understand how soon gradual increases in temperature over this region would be discernible from the year-to-year variation.

1.1 United States Historical Climatology Network

Recently the U. S. Carbon Dioxide Information Analysis Center established a network of 1219 stations (the HCN network) for the contiguous United States "with the objective of compiling a data-set suitable for the detection of climatic change" (Quinlan, Karl & Williams (1987)). The network record includes maximum, minimum, mean monthly temperatures and total monthly precipitation since 1890s.

As part of a study to improve the land surface parameterization of the GFDL-GCM (Geophysical Fluid Dynamics Laboratory - Global Climate Model), a 41 year daily value database was prepared and made available in CDROM format. (Wallis, Lettenmaier and Wood (1991)). The study used 1036 of the original HCN stations, with missing days flagged and then estimated by correlation to nearby stations. There are differences between the unadjusted monthly mean values reported by Quinlan *et al* and those reported by Wallis *et al*. Some of these differences can be accounted for by a difference in the treatment of missing days; Quinlan *et al* summed the observed days in any given month and divided by their number to calculate the average monthly value; missing days were not estimated. However, Wallis *et al* encountered many cases where even the number of missing days in a given month did not agree between the daily and monthly NCDC data bases. Sites used in our study were chosen so as to minimize the effect of these data peculiarities. If the National Climate Data Center ever prepares a cleaned-up data base, then the calculations reported here could easily be seen. In the interim, we do not believe that data errors for the stations and periods used in this study are large enough to invalidate any of our results or conclusions.

2. ANALYSIS OF SPATIAL STRUCTURE

In this section we discuss a spatial model appropriate for a meteorological field over a single time period. The field discussed here is the average winter temperature. The daily average temperature at a location is defined to be the mean of the daily maximum and the daily minimum at that location. The average winter temperature

is defined to be the average daily average temperature over the months December, January and February. The generalization to include a temporal component is the subject of the following section.

Figure 1 represents the region under study. The locations of the 88 U. S. historical climatological network stations in the region are marked with '+'. The elevation of the stations, in feet, are represented by the overlaying gray-scale image.

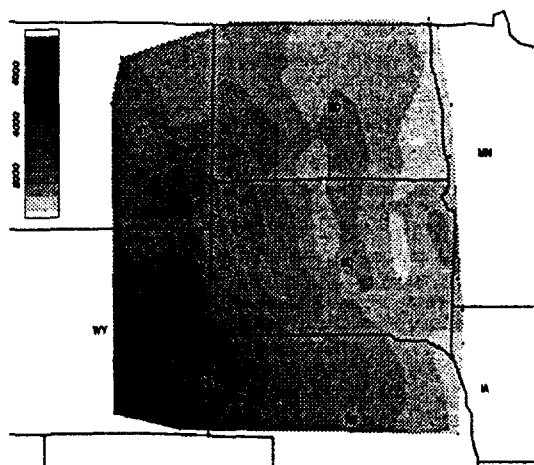


FIGURE 1: Locations and elevations of the U. S. historical climatological network stations. The gray-level image is for the elevation above sea-level over the region.

Note that the elevation increases from east to west and also from north to south. As expected, the elevation of the station has a marked impact on temperature and it is essential that the model reflect this relationship. Figure 2 is a gray-level image of the mean winter temperatures for the winter of 1983-84. The winter of 1983-84 is the basis for the examples in this section.

Suppose $Z(x)$ is a real-valued stationary Gaussian random field on the region under study (R) with mean

$$E\{Z(x)\} = f'(x)\beta,$$

where $f(x) = \{f_1(x), \dots, f_q(x)\}'$ is a known vector function, and β is a vector of unknown regression coefficients. Furthermore, the covariance function is represented by

$$\text{cov}\{Z(x), Z(y)\} = \alpha K_\theta(x, y) \quad \text{for } x, y \in R$$

where $\alpha > 0$ is a scale parameter, $\theta \in \Theta$ is a $q \times 1$ vector of structural parameters and Θ is an open set in \mathbb{R}^p . The division is purely formal as θ may also determine aspects of scale. This formulation is standard for meteorological networks (Gandin (1963)).

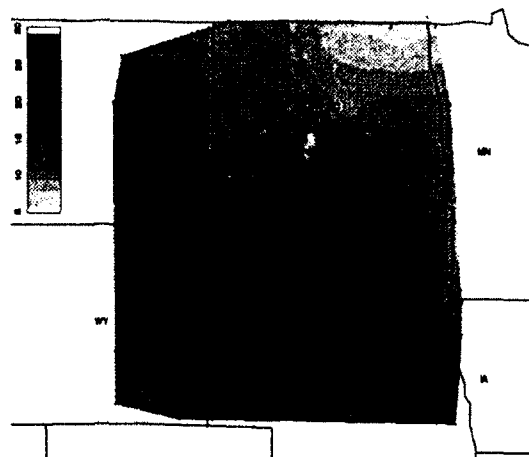


FIGURE 2: Graylevel image of the mean winter temperatures for the winter of 1983-84. The temperature range, in Fahrenheit, is given in the legend.

We observe, from a single realization of the field, $\{Z(x_1), \dots, Z(x_n)\}' = Z$ where x_1, \dots, x_n are the spatial locations of the stations in the network. We will focus on the prediction of $Z(x_0)$, where x_0 is a new location in the region of interest.

The Kriging predictor is the best linear unbiased predictor of the form $\hat{Z}_\theta(x_0) = \lambda'(\theta)Z$, that is, the unbiased linear combination of the observations that minimizes the variance of the prediction error. The quality of the prediction is determined by the distribution of the prediction error, $e(x_0) = Z(x_0) - \hat{Z}_\theta(x_0)$. Note that the underlying Kriging procedure is motivated by sampling considerations, producing point predictions and associated measures of uncertainty for those predictions both based on sampling distributions unconditional on the observed Z . However it is well known that Kriging, when the mean is of known regression form, can be given a Bayesian interpretation (see e.g., Omre & Halvarsen (1989), Handcock & Stein (1989), Hastie and Tibshirani (1990)).

The implementation of this model requires the specification of the regression function $f(x)$ and the spatial covariance structure $K_\theta(\cdot, \cdot)$. The regression function is composed of easily measurable spatial characteristics of the station such a latitude, longitude, elevation and distance to the closest urban area. We shall explore these choices in §3. The specification of the covariance structure is dealt with in the following section.

2.1 Spatial Correlation Structure

In this section we describe a general class of isotropic and homogeneous covariance functions that we feel provides a sound foundation for the parametric modeling of Gaussian random fields. An isotropic and homogeneous covariance function can be represented as $K(x, y) \equiv K(|x - y|)$, $\forall x, y \in \mathbb{R}$, so that the class is usually written as a function of a single scalar variable $K(x)$, $x \in \mathbb{R}$. The class is motivated by the smooth nature of the spectral density, the wide range of behaviors covered and the interpretability of the parameters (See Handcock & Stein (1989), Matérn (1986)). The isotropic covariance functions have the form

$$K_{\theta}(x) = \frac{1}{2^{\theta_2-1}\Gamma(\theta_2)} \cdot \left(\frac{2\sqrt{\theta_2} \cdot x}{\theta_1}\right)^{\theta_2} \mathcal{K}_{\theta_2}\left(\frac{2\sqrt{\theta_2} \cdot x}{\theta_1}\right)$$

where $\theta_1 > 0$ is a spatial scale parameter controlling the range of correlation and $\theta_2 > 0$ is the parameter controlling the smoothness of the field. A field with this covariance function is $\lceil \theta_2 - 1$ times (mean-square) differentiable (Cramér & Leadbetter (1967)). Here \lceil is the gamma function, \lceil is the integer ceiling function and \mathcal{K}_{θ_2} is the modified Bessel function of the third kind and order θ_2 discussed in Abramowitz & Stegun (1964), §9.

The well known "Exponential" class corresponds to the sub-class with smoothness parameter $\theta_2 = 1/2$, that is

$$K_E(x) = \theta_1 \exp(-x/\theta_1).$$

We call this class the Matérn class because of the general treatment given in the seminal work of Matérn (1986).

The application in this paper has only two spatial dimensions ($d = 2$) leading to some simplification.

2.2 Model Development and Validation

In traditional Kriging, one estimates α and θ by either likelihood methods or various *ad hoc* approaches. The likelihood approach to the estimation of the covariance structure was first applied in the hydrological and geological fields following Kitanidis (1983), Kitanidis & Lane (1985) and Hoeksema & Kitanidis (1985). See also Mardia & Marshall (1984), Handcock & Stein (1989), and Handcock (1989). Usually the predictor and the behavior of the prediction error are themselves estimated by 'plugging-in' the estimates to the expressions for known α and θ .

The values of α , θ and β that maximize this log-likelihood are denoted by, $\hat{\alpha}$, $\hat{\theta}$ and $\hat{\beta}$, respectively.

3. ANALYSIS OF TEMPORAL STRUCTURE

In this section we consider the temporal component of the model, generalizing the random field to $Z_t(x)$ where $t = 1937, \dots$ represents the winter of observation. We consider the time-series of data from each station, independent of the spatial information.

The time-series of a single site exhibits little short-term or long-term dependence. Figure 3 presents the time-series and empirical autocorrelation functions for four spatially separate stations. Individually the time-series are quite variable overtime. The right hand side figures are the sample autocorrelation functions corresponding to the time-series. The dashed boundaries represent approximate 95% confidence limits. Note the similar patterns in the series over time and the lack of first lag autocorrelation. The median cross-correlation is 0.78 with quantiles 0.64 and 0.87, indicating a tendency to move in conjunction with each other over time and reflecting the influence of the spatial correlation modeled in the previous section.

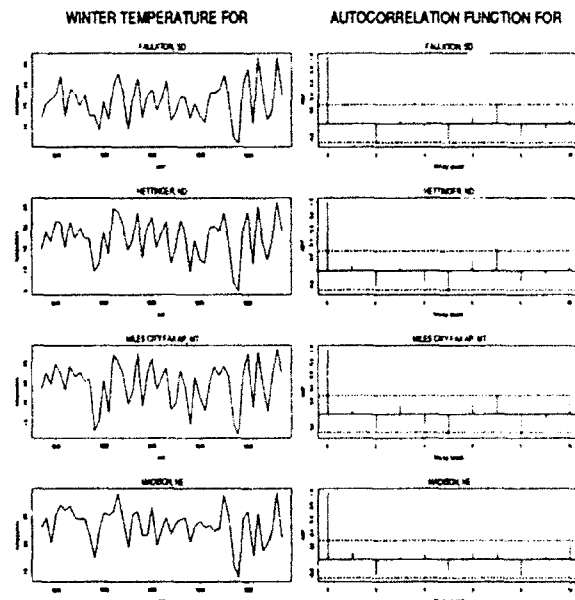


FIGURE 3: Time series and empirical autocorrelation functions for four typical sites.

Our analysis indicates that there is little evidence of either short-memory or long-memory dependency in the time-series of the stations. There is little evidence that the spatial structure changes over the time scale considered here. The substance of the analysis is given in Handcock & Wallis (1990).

4. MEASURING AREAL MEAN TEMPERATURE

In the previous sections we found a complex spatial structure to the mean winter temperatures, little temporal dependence structure and little evidence for changing spatial structure over time. In this section we focus interest in a measure of the areal mean temperature over the region of interest. The time-series of areal mean temperatures is defined by:

$$\bar{Z}_t = \frac{1}{|R|} \int_R Z_t(x) dx \quad t = 1937, \dots, 1986, \dots$$

where $|R|$ is the area of the region R . Thus at each point in time, \bar{Z}_t represents the average temperature over the region and is a function of the field $Z(x)$. \bar{Z}_t provides a natural measure for the detection of changing climatic patterns over the region. As the region is devoid of gross topographic features, it provides a convenient measure of overall temperature during the winter. It is important to note that \bar{Z}_t is a characteristic of the temperature field itself, and not a characteristic of the stations in the network. The behavior of the areal mean temperature will provide an indication of the overall changes in climate over the region independent of the individual stations.

Based on our model, we can summarize the available information for \bar{Z}_t from the predictive density $P(\bar{Z}_t | Z_{1937}, Z_{1938}, \dots, Z_{1986})$, that is, the posterior density of \bar{Z}_t given the complete spatial-temporal information available.

How can we further summarize the areal mean temperature? The distributions are symmetric and have a similar t-like distributional shape. The ratio of largest to smallest variance is 2.6. To further explore the temporal changes in \bar{Z}_t we will consider the time-series of maximum *a posteriori* (MAP) values. While this clearly represents a reduction in information relative to the full distribution, it facilitates examination.

Figure 4 represents the MAP values for the last half century. Note the lack of a clear trend over time. Some interesting years have been indicated. The last year for which data are available (1986) also has the highest temperature.

A reasonable model for the mean areal temperature over the last half century is

$$\bar{Z}_t = \mu_t + \epsilon_t \quad t = 1937, \dots, 1986, \dots \quad (4.1)$$

where $\{\epsilon_t\}_{t=1937}^{1986}$ is an independent, and identically distributed Gaussian sequence with zero mean and variance σ^2 . The sequence $\{\mu_t\}_{t=1937}^{1986}$ represents the mean level. The motivation is the absence of strong temporal dependence (§4.1) and the approximate constant variances.

The base line model is that the means are temporally stable: $\mu_t \equiv \mu$. Here μ will be called the static areal mean temperature. The details of the model are given in Hancock & Wallis (1991).

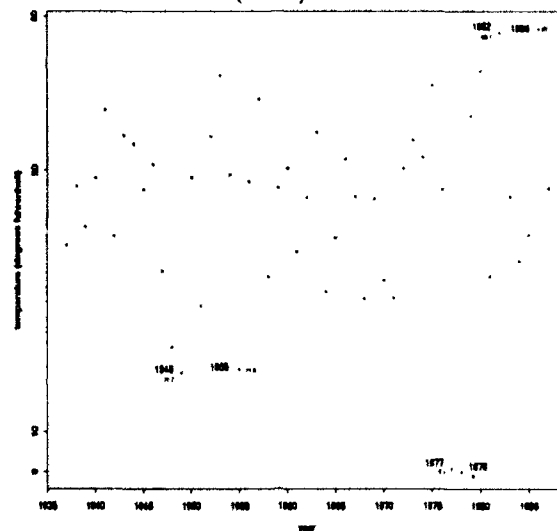


FIGURE 4: This is the time-series of (the MAP estimates for the) areal mean temperature. There is little short or long term persistence. The least square line, which has a non-significant slope, is marked.

5. CALIBRATING CHANGES IN MEAN AREAL TEMPERATURE

The primary motivation for developing these models is as a tool to calibrate changes in the mean areal temperature. The model in the previous section allows this to be done. Many scenarios have been proposed for future global warming. Typically their basis is mathematical rather than empirical.

Consider the distributions in Figure 5. The solid line is the posterior distribution for the static areal mean temperature from 1937-86, this should be compared to Figure 6. It summarizes our uncertainty about the static areal mean temperature by a distribution with mean about $19.7^\circ F$ and a standard deviation of $0.51^\circ F$.

How soon could a $5^\circ F$ gradual increase in static areal mean temperature in this region over half a century be discernible?

Suppose we collected data from the network in this region for the next ten years, as the underlying static temperature gradually increases. Suppose that, besides the creep in level, the stochastic structure remained stable, as is supported by the analysis in §4. The dashed posterior (marked by '10') represents a hypothetical posterior for the static areal mean temperature based on

that 10 years of data. Note that there is still marked overlap with the summary from the last 50 years (solid line). This indicates that it would be extremely difficult to discern such a gradual increase after only ten years.

Also plotted are the hypothetical posteriors after collecting 20, 30 and 50 years of information. Only after about 20-30 years is the gradual level increase discernible with high probability. If additional years of data were incorporated, the scale of the posterior based on the historical record will decrease. That is, the hypothetical experiment conducted here is conservative.

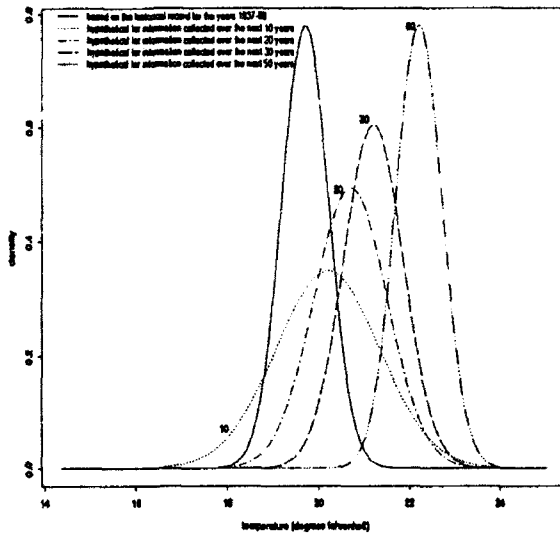


FIGURE 5: The predictive distribution for the static areal mean temperature for 1937-86. The dashed distributions represent hypothetical posterior knowledge for the static areal mean temperature based on additional years of data.

6. CONCLUSION

This approach facilitates the calibration of changes in the areal mean temperature against the historical record. For the scenario of a gradual increase of 5°F over 50 years, it will take 20-30 years of data before the change will be discernible from the natural variation in temperatures. By the time the necessary information had accumulated an increase of between $2 - 3^{\circ}\text{F}$ would already have occurred. The application of the model to alternative global warming scenarios is straight forward. There is no indication that the areal mean temperature for this time of the year in this region has changed over the last half century.

Similar models investigating summer temperatures and precipitation for this region will be reported elsewhere. These facilitate the measurement of runoff and

hence address the central issue of the evaluation of water resources.

REFERENCES

- ABRAMOWITZ, M. & STEGUN, I. A. (1965). *Handbook of Mathematical Functions*. New York: Dover.
- CRAMÉR, H. & LEADBETTER, M. R. (1967). *Stationary and Related Stochastic Processes*. New York: Wiley.
- DIAZ, H. F. & QUAYLE, R. G.. (1978). The 1976-77 winter in the contiguous United States in comparison with past records *Monthly Weather Review*, 106, 1393-1421.
- DIAZ, H. F. & QUAYLE, R. G.. (1980). The Climate of the United States since 1895: Spatial and Temporal Changes. *Monthly Weather Review*, 108, 249-266.
- GANDIN, L. S. (1963; 1965). *Objective Analysis of Meteorological Fields*. GIMIZ, Leningrad, 238p. Translated from Russian by Hardin, R., Israel program for Scientific Translations, Jerusalem, 242p.
- HANDCOCK, M. S. (1989). Inference for Spatial Gaussian Random Fields when the Objective is Prediction. Ph.D. thesis, Department of Statistics, University of Chicago, Chicago, Illinois.
- HANDCOCK, M. S. and STEIN, M. L. (1989). A Bayesian analysis of Kriging. Submitted to *Technometrics*.
- HANDCOCK, M. S. and WALLIS, J. R. (1991). An Approach to Statistical Spatial-Temporal Modeling of Meteorological Fields. Submitted to *JASA* and presented at the conference *European Geophysical Society XV General Assembly*, held in Copenhagen, Denmark on April 23-27, 1990.
- HOEKSEMA, R. J. & KITANIDIS, P. K. (1985). Analysis of the Spatial Structure of Properties of Selected Aquifers. *Water Resources Research* 21, 563-572.
- INTERGOVERNMENTAL PANEL ON CLIMATE CHANGE, (1990) *Scientific Assessment of Climate Change*, World Meteorological Organization, United Nations Environment Programme, Geneva, Switzerland.
- JONES, P. D., RAPER, S. C. B., BRADLEY, R. S., DIAZ, H. F., KELLY, P. M., & WIGLEY, T. M. L. (1986). Northern Hemisphere Surface Air Temperature Variations 1851-1984. *Journal of Climate and Applied Meteorology* 25, 161-179.
- JONES, R. H. (1989). Fitting a Stochastic Partial Differential Equation to Aquifer Head Data. *Stochastic Hydrology and Hydraulics* 3, 100-105.
- KARL, T. R., LIVEZEY, R. E. & EPSTEIN, E. S. (1984). Recent Unusual Mean Winter Temperatures Across

- the Contiguous United States. *Bull. Amer. Meteor. Soc.*, **65**, 1302-1309
- KARL, T. R. (1985). Perspective on Climate Change in North America during the Twentieth Century. *Physical Geography*, **6**, 207-229
- KARL, T. R., WILLIAMS, C. M. YOUNG, P. J. & WENDLAND, W. M. (1986). A Model to Estimate the Time of Observation Bias associated with monthly mean maximum, minimum and mean temperatures in the United States. *Journal of Climate and Applied Meteorology* **25**, 145-160
- KARL, T. R., HEIM, R. R. & QUAYLE, R. G. (1991). The Greenhouse Effect in Central North America: If Not Now, When? *Science*, **251**, 1058-1061
- KITANIDIS, P. K. (1983). Statistical Estimation of Polynomial Generalized Covariance Functions and Hydrologic Applications. *Water Resources Research* **19**, 909-921.
- KITANIDIS, P. K. & LANE, R. W. (1985). Maximum Likelihood Parameter Estimation of Hydrologic Spatial Processes by the Gauss-Newton Method. *Journal of Hydrology* **17**, 31-56.
- LETTENMAIER, D. P., WOOD, E. F. & WALLIS, J. R. (1991). A Daily Hydroclimatological Data Set for the Continental United States. *Water Resources Research* **27**, 7, 1657-1663.
- MARDIA, K. V. & MARSHALL, R. J. (1984). Maximum likelihood estimation of models for residual covariance in spatial regression. *Biometrika* **71**, 135-146.
- MATHERON, G. (1965). *Les Variables Régionalisées et leur Estimation* Paris: Masson.
- MATÉRN, B. (1986). *Spatial Variation* Second Ed, *Lecture Notes in Statistics*, **36**. Berlin: Springer-Verlag.
- MITCHELL, J. F. B. (1989). The "Greenhouse Effect" and Climate Change. *Reviews of Geophysics* **27**, 115-139
- QUINLAN, F. T., KARL, T. R. & WILLIAMS, C. N., Jr. (1987). *United States Historical Climatology Network (HCN) serial temperature and precipitation data*. NDP-019, Carbon Dioxide Information Analysis Center, Oak Ridge National Laboratory, Oak Ridge, Tennessee.

Space-Time Models for the Analysis of Satellite Temperature and Ozone Data*

Xufeng Niu

Department of Statistics
Florida State University
Tallahassee, FL 32306

George C. Tiao

Graduate School of Business
University of Chicago
Chicago, IL 60637

Abstract

A class of space-time bilateral models is developed for satellite ozone and temperature data. In the spatial direction, both latitudinal and longitudinal interactions are considered. It is shown that after some appropriate transformations, the space-time bilateral models can be expressed as temporal vector processes with a small number of natural parameters. Covariance structures and stationarity conditions of the error terms in the models are studied. Using properties of circular matrices, likelihood functions for the parameters in the models are derived. Applications of the space-time bilateral models to TOMS (Total Ozone Mapping Spectrometer) ozone data are discussed.

1 Introduction

In recent years, a decline in column ozone over much of the earth and the possibility of global greenhouse warming due to the rapid increase in radiatively active atmospheric trace gases are receiving increased attention. Satellite ozone and temperature data, measured by the Total Ozone Mapping Spectrometer (TOMS) and the Solar Backscattered Ultraviolet (SBUV) instruments, respectively, play an important role in assessing the global changes. For instance, the GRIDTOMS ozone tapes processed by the National Aeronautics and Space Administration (NASA) provide daily total column ozone data on each of 1° latitude by 1.25° longitude cells, which enable us to estimate the changes in ozone as a function of geographical location.

*Computations for this paper were performed using computer facilities in the Department of Statistics and Statistical Consulting Center at the Florida State University. We thank the Ozone Processing Team of NASA/Goddard Space Flight Center for providing us with the GRIDTOMS data. The second author was supported by National Aeronautics and Space Administration grant number NAGW-2057.

Besides the satellite measurements, there are more than 100 ground stations that observe daily total column ozone and over 200 stations around the world that collect temperature data. For the ground-based ozone and temperature data, one of the widely used models is the regression-time series model. Letting $y(t)$ denote the monthly average of total ozone at time t over a particular station, Reinsel et al. (1981) described the regression-time series model as follows:

$$\begin{cases} y(t) = \mu + S(t) + \omega R(t) + \xi(t) \\ \xi(t) = \sum_{i=1}^p \phi_i \xi(t-i) + \epsilon(t), \end{cases} \quad (1.1)$$

where μ is an overall level, $S(t)$ is a seasonal component, adequately represented by a linear combination of sinusoidal curves of fundamental period 12 and their harmonics (e.g., 6 months), and $\omega R(t)$ represents a linear trend with

$$R(t) = \begin{cases} (t - t_0)/12 & \text{if } t \geq t_0, \\ 0 & \text{if } t < t_0, \end{cases}$$

where t_0 corresponds to January 1970. The residual series $\xi(t)$ was modeled as an autoregressive process $AR(p)$, where p usually is equal to 1, and $\epsilon(t)$ is a process of independent random variables with mean zero and possibly different variance for different months of the year. Some other covariates, such as solar flux series, may be included in the model. Reinsel et al. (1988) also used model (1.1) to analyze the 7-year record of SBUV satellite ozone data on 10° latitude by 20° longitude cells.

Satellite data form space-time dependent lattice systems since the observations on the cells are related to one another and the time. Understanding and describing the statistical structures of these space-time lattice systems are very important. Spatial lattice systems have been studied by many researchers, since the analysis of spatial series is of interest in a number of fields such

as geography, econometrics, geology and ecology. Whittle (1954), a fundamental paper on this topic, discussed some bilateral models for the processes in a plane. For the one dimensional case, denoting observations along a line by ξ_i ($i = \dots, -2, -1, 0, 1, 2, \dots$) and error terms by ϵ_i ($i = \dots, -2, -1, 0, 1, 2, \dots$), respectively, Whittle's first-order bilateral autoregressive model has the form

$$\xi_i = \alpha_1 \xi_{i-1} + \alpha_2 \xi_{i+1} + \epsilon_i.$$

In a sense this model may be regarded as a simple generalization of a first-order autoregressive AR(1) model, but it presents some peculiarities which makes its analysis more difficult. The essential difference is that in a time series we have the natural distinction of past and future, and the value of the current observation depends only upon the past values. However, for a spatial line process, there is no such distinction between the two directions, and dependence should naturally extend both ways. Suppose that the error term ϵ_i are independently and identically normally distributed. Then unlike the time series AR(1) case, an attempt to estimate the coefficients α_1 and α_2 by minimizing $\sum_i (\xi_i - \alpha_1 \xi_{i-1} - \alpha_2 \xi_{i+1})^2$ leads to nonsensical results (see Whittle (1954) for details). For an infinite two dimensional spatial lattice system, it is usually not possible to find a simple formula for the likelihood function for the parameters in a bilateral lattice model, since the Jacobian of the transformation from the error terms to the observations will depend upon the unknown parameters in a complicated way. In order to find a sensible way to estimate the parameters in a bilateral lattice system, Whittle (1954) constructed a unilateral representation of a bilateral lattice model such that the unilateral model and the bilateral model generate the same spectral function. For the one dimensional case, the unilateral representation of a finite bilateral autoregressive model is also a finite autoregression, and the estimates of the parameters in a bilateral model can be calculated from the estimates of the parameters in its unilateral representation. However, for the two dimensional case, most of the finite-order bilateral lattice models have infinite half-space unilateral representations, which are so complicated that nothing is gained by performing the transformation.

Due to the difficulties of estimating the parameters in a bilateral model, a great deal of efforts have been made to develop unilateral models for spatial lattice systems. Besag (1972) discussed the correlation structure of half-space unilateral autoregressive stationary processes for the two dimensional case. Tjøstheim (1978) introduced unilateral causal (quadrant-type) models for high dimensional lattice systems. For the two dimensional case, suppose that $\{\xi_{i,j}\}$ is a random spatial series and $\{\epsilon_{i,j}\}$

a white noise innovation series where (i, j) varies over a regular cartesian lattice. Then Tjøstheim's first-order unilateral autoregressive and moving-average (ARMA) spatial series model can be written as

$$\xi_{i,j} + \alpha_{10}\xi_{i-1,j} + \alpha_{01}\xi_{i,j-1} + \alpha_{11}\xi_{i-1,j-1} = \epsilon_{i,j} + \theta_{10}\epsilon_{i-1,j} + \theta_{01}\epsilon_{i,j-1} + \theta_{11}\epsilon_{i-1,j-1}. \quad (1.2)$$

Similar to time series analysis, Tjøstheim (1978) studied the stability and invertibility conditions for the unilateral ARMA spatial series models, and presented a Yule-Walker type estimates for the parameters in a unilateral AR spatial series model. He also pointed out in the paper that under mild conditions the estimates are consistent and asymptotically normal. As a continuation of this paper, Tjøstheim (1981) considered some possible applications of these models; then (1983) showed that the asymptotic results for the unilateral causal models are also valid for the half space spatial lattice models.

A basic feature of satellite temperature and ozone data is their circular property, i.e., for a fixed latitude and a fixed time t , the data are observed along a circle. Niu (1991) introduced a class of space-time bilateral autoregressive and moving average models for the satellite ozone data on a fixed latitude, and discussed some basic properties of the models. For a fixed latitude, let $y_j(t)$ be the monthly average ozone observations at longitude j . Niu considered an additive model:

$$y_j(t) = S_j(t) + R_j(t) + \xi_j(t) \quad (1.3)$$

where $S_j(t)$ is a seasonal component, $R_j(t)$ a trend component, and $\xi_j(t)$ a noise component. For the noise term $\xi_j(t)$, he modeled it in the following space-time form:

$$\begin{aligned} \xi_j(t) = & \sum_{k=1}^{p_1} [\alpha_k \xi_{j-k}(t) + \theta_k \xi_{j+k}(t)] \\ & + \sum_{l=1}^{p_2} \phi_l \xi_j(t-l) + \epsilon_j(t) \\ & - \sum_{k=1}^q [\nu_k \epsilon_{j-k}(t) + \mu_k \epsilon_{j+k}(t)], \end{aligned} \quad (1.4)$$

$$j = 1, 2, \dots, n; \quad t = 1, 2, \dots, T$$

where $\epsilon_j(t)$'s are assumed to be independent and normally distributed random variables with mean zero and variance $\sigma^2(t) = \sigma^2(t-12)$. The model in (1.3) and (1.4) were applied to the TOMS ozone data to assess the long term trends.

In this study, we extend the space-time models for satellite ozone and temperature data introduced by Niu

(1991). Instead of modeling the data for each latitude separately, both latitudinal and longitudinal interactions are considered. Specifically, let $y_{lj}(t)$ be the monthly average ozone or temperature observations at latitude l and longitude j . We consider an additive model for $\{y_{lj}(t), t = 1, 2, \dots, T\}$ similar to the model in (1.3).

$$y_{lj}(t) = S_{lj}(t) + R_{lj}(t) + \xi_{lj}(t) \quad (1.5)$$

The noise term $\xi_{lj}(t)$ is modeled in the following space-time form:

$$\begin{aligned} \xi_{lj}(t) = & \sum_{k=1}^{p_1} \alpha_{1k1} \xi_{l,j-k}(t) + \alpha_{1k2} \xi_{l,j+k}(t) \\ & + \theta_{l-1} \xi_{l-1,j} + \theta_l \xi_{l+1,j} \\ & + \sum_{k=1}^{p_2} \phi_{lk} \xi_{lj}(t-k) + \epsilon_{lj}(t) \end{aligned} \quad (1.6)$$

$$l = 1, \dots, m; \quad j = 1, 2, \dots, n; \quad t = 1, 2, \dots, T.$$

Again, $\epsilon_{lj}(t)$'s are assumed to be independent and normally distributed random variables with mean zero and variance $\sigma_l^2(t) = \sigma_l^2(t-12)$. We call this model a space-time autoregressive (STAR($p_1, 1, p_2$)) model.

In section 2, we discuss some basic properties of the space-time models. A definition of block circular matrices will be introduced, which will be used to describe the covariance structures of the error term ξ_{lj} . The stationarity conditions of the spatial process are examined, and the likelihood function for the parameters in some specific space-time models is derived. In section 3, the space-time models are applied to TOMS ozone data. The statistical structure of the de-seasonalized and detrended series will be studied. Finally, conclusions are summarized in section 4.

2 Some basic properties of the models

In this section, we discuss basic properties of the space-time models in (1.6). For simplicity, we consider only the first order space-time model for the error term, i.e.,

$$\begin{aligned} \xi_{lj}(t) = & \alpha_{11} \xi_{l,j-1}(t) + \alpha_{12} \xi_{l,j+1}(t) \\ & + \theta_{l-1} \xi_{l-1,j} + \theta_l \xi_{l+1,j} \\ & + \phi_1 \xi_{lj}(t-1) + \epsilon_{lj}(t) \end{aligned} \quad (2.1)$$

$$l = 1, \dots, m; \quad j = 1, 2, \dots, n; \quad t = 1, 2, \dots, T.$$

To describe the covariance structure of this model, we first introduce the definition of block circular matrices.

2.1 Block circular matrices

The definition of circular matrices was given by Good (1954).

Definition 2.1 An $n \times n$ real matrix C is called a *circular* if it has the form

$$C = \{c_{j-i}\}, \quad i = 0, 1, \dots, n-1; \quad j = 0, 1, \dots, n-1$$

where the suffices are reduced mod n , i. e., $c_{-k} = c_{n-k}$.

For example, a symmetrical circular matrix with $n = 5$ has the form

$$C = \begin{bmatrix} c_0 & c_1 & c_2 & c_2 & c_1 \\ c_1 & c_0 & c_1 & c_2 & c_2 \\ c_2 & c_1 & c_0 & c_1 & c_2 \\ c_2 & c_2 & c_1 & c_0 & c_1 \\ c_1 & c_2 & c_2 & c_1 & c_0 \end{bmatrix}.$$

Good (1954) studied properties of circulars in detail. One of the main properties of the circulars is the following:

Lemma 2.1 If C and D are circulars, then so are $C + D$, $C - D$, CD and any polynomial in C . Furthermore, if C^{-1} exists, then C^{-1} is also a circular. Let $\{v_s(C) : s = 1, \dots, n\}$ and $\{v_s(D) : s = 1, \dots, n\}$ be the eigenvalues of the matrices C and D , then the eigenvalues of $C + D$, $C - D$ and CD are respectively

$$v_s(C) + v_s(D), \quad v_s(C) - v_s(D), \quad v_s(C)v_s(D)$$

Now let us examine some special circulars. Define an $n \times n$ matrix

$$W = \begin{bmatrix} 0 & 0 & 0 & \dots & \dots & 0 & 0 & 1 \\ 1 & 0 & 0 & \dots & \dots & 0 & 0 & 0 \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ 0 & 0 & 0 & \dots & \dots & 0 & 1 & 0 \end{bmatrix}.$$

Then W is orthogonal and circular. Notice that premultiplying an $n \times n$ matrix A by W causes changing the last row of A into its first row, the first row of A into its second row, and so on. Hence it is easy to see that $W^n = I_n$. Since $W^{n-k}W^k = I_n$, we have $W^{n-k} = W^{-k}$. Furthermore, the matrices $\{W^k, k = 0, \pm 1, \pm 2, \dots\}$ are commutative. Since the eigenvalues of W , $\{e^{i2s\pi/n}, s =$

$1, 2, \dots, n\}$, are distinct, there exists a nonsingular matrix P_n , such that

$$\Lambda = P_n^{-1} W P_n = \begin{bmatrix} e^0 & 0 & 0 & \dots & 0 \\ 0 & e^{i2\pi/n} & 0 & \dots & 0 \\ \dots & \dots & \dots & \dots & \dots \\ 0 & 0 & 0 & \dots & e^{i2(n-1)\pi/n} \end{bmatrix}.$$

Thus,

$$P_n^{-1} W^k P_n = \Lambda^k, \quad k = \pm 1, \pm 2, \dots, \pm n.$$

Similar to the definition of circular matrix, we give the definition for a block circular matrix.

Definition 2.2 Suppose that an $mn \times mn$ matrix C is composed of an $n \times n$ matrix of $m \times m$ submatrices. We call C a block circular matrix if it has the form

$$C = \{C_{j-i}\}, \quad i = 0, 1, \dots, n-1; \quad j = 0, 1, \dots, n-1$$

where C_k 's are $m \times m$ matrices and the suffices are reduced mod n , i.e., $C_{-k} = C_{n-k}$.

Notice that in this definition, the submatrices C_{j-i} , ($i = 0, 1, \dots, n-1; j = 0, 1, \dots, n-1$), themselves may not be circulices. If C and D are block circulices, it is easy to see that so are $C+D$ and $C-D$. Let $A = (a_{jk})$ and $B = (b_{jk})$ be $m \times m$ and $n \times n$ matrices, respectively. We denote the Kronecker product of A and B by

$$A \otimes B = (a_{jk} B).$$

A general block circular matrix C can be expressed in terms of W as

$$\begin{aligned} C &= I_n \otimes C_0 + (W \otimes C_{n-1} + W^{-1} \otimes C_1) \\ &\quad + (W^2 \otimes C_{n-2} + W^{-2} \otimes C_2) + \dots \\ &\quad + \left(W^{\frac{n-1}{2}} \otimes C_{\frac{n+1}{2}} + W^{-\frac{n-1}{2}} \otimes C_{\frac{n-1}{2}} \right), \\ &\quad \text{for } n \text{ odd} \end{aligned} \quad (2.2)$$

and

$$\begin{aligned} C &= I_n \otimes C_0 + (W \otimes C_{n-1} + W^{-1} \otimes C_1) \\ &\quad + (W^2 \otimes C_{n-2} + W^{-2} \otimes C_2) + \dots \\ &\quad + \left(W^{\frac{n-2}{2}} \otimes C_{\frac{n+2}{2}} + W^{-\frac{n-2}{2}} \otimes C_{\frac{n-2}{2}} \right) \\ &\quad + W^{\frac{n}{2}} \otimes C_{\frac{n}{2}}, \quad \text{for } n \text{ even.} \end{aligned} \quad (2.3)$$

From this expression, we can show that if C and D are block circulices, then so are CD and any polynomial in C . Furthermore, we have the following result:

Lemma 2.2 Suppose that C_0, C_1, \dots, C_p and D_1, D_2, \dots, D_p are some $m \times m$ matrices, and

$$C = I_n \otimes C_0 + \sum_{k=1}^p (W^k \otimes C_k + W^{-k} \otimes D_k).$$

Then

$$\begin{aligned} (P_n^{-1} \otimes I_m) C (P_n \otimes I_m)' &= \\ I_n \otimes C_0 + \sum_{k=1}^p (\Lambda^k \otimes C_k + \Lambda^{-k} \otimes D_k). \end{aligned}$$

i.e., the block circular matrix C is similar to a diagonal block matrix with diagonal block elements

$$C_0 + \sum_{k=1}^p \left(e^{i2sk\pi/n} C_k + e^{-i2sk\pi/n} D_k \right),$$

$$s = 0, 1, \dots, n-1.$$

2.2 Stationarity conditions for the noise term

In order to derive the stationarity conditions for the noise term in model (2.1), we now express the model of $\{\xi_{ij}(t)\}$ in terms of a vector temporal process. Define

$$\begin{aligned} \xi_j(t) &= [\xi_{1j}(t), \xi_{2j}(t), \dots, \xi_{mj}(t)]', \\ \epsilon_j(t) &= [\epsilon_{1j}(t), \epsilon_{2j}(t), \dots, \epsilon_{mj}(t)]', \\ j &= 1, 2, \dots, n; \quad t = 1, 2, \dots, T. \end{aligned}$$

$$B = \begin{bmatrix} 1 & -\theta_1 & 0 & \dots & 0 & 0 \\ -\theta_1 & 1 & -\theta_2 & \dots & 0 & 0 \\ 0 & -\theta_2 & 1 & \dots & 0 & 0 \\ \dots & \dots & \dots & \dots & \dots & \dots \\ 0 & 0 & 0 & \dots & 1 & -\theta_{m-1} \\ 0 & 0 & 0 & \dots & -\theta_{m-1} & 1 \end{bmatrix}.$$

and

$$A_1 = \text{Diag}(\alpha_{11}), \quad A_2 = \text{Diag}(\alpha_{12}), \quad \Phi = \text{Diag}(\phi_1).$$

Here B, A_1, A_2 and Φ are all $m \times m$ matrices. Now the model in (2.1) can be written as:

$$\begin{aligned} B\xi_j(t) &= A_1\xi_{j-1}(t) + A_2\xi_{j+1}(t) + \Phi\xi_j(t-1) \\ &\quad + \epsilon_j(t). \end{aligned} \quad (2.4)$$

Further, let

$$\begin{aligned} \varphi(B, W) &= I_n \otimes B - (W \otimes A_1 + W^{-1} \otimes A_2), \\ \xi(t) &= [\xi'_1(t), \xi'_2(t), \dots, \xi'_n(t)]', \\ \epsilon(t) &= [\epsilon'_1(t), \epsilon'_2(t), \dots, \epsilon'_n(t)]', \\ t &= 1, 2, \dots, T. \end{aligned}$$

Then both $\xi(t)$ and $\epsilon(t)$ are $n \times m$ random vectors, and the model in (2.4) can further be written in a form:

$$\varphi(B, W)\xi(t) = (I_n \otimes \Phi)\xi(t-1) + \epsilon(t). \quad (2.5)$$

By Lemma 2.2, the matrix $\varphi(B, W)$ is similar to a diagonal block matrix with the diagonal block elements

$$A_{s,m} = B - e^{i2s\pi/n} A_1 - e^{-i2s\pi/n} A_2,$$

$$= \begin{bmatrix} \lambda_{1s} & -\theta_1 & 0 & \dots & 0 & 0 \\ -\theta_1 & \lambda_{2s} & -\theta_2 & \dots & 0 & 0 \\ 0 & -\theta_2 & \lambda_{3s} & \dots & 0 & 0 \\ \dots & \dots & \dots & \dots & \dots & \dots \\ 0 & 0 & 0 & \dots & \lambda_{m-2,s} & -\theta_{m-1} \\ 0 & 0 & 0 & \dots & -\theta_{m-1} & \lambda_{ms} \end{bmatrix}$$

$$s = 0, 1, \dots, n-1$$

where

$$\lambda_{ls} = 1 - \alpha_{11}e^{i2s\pi/n} - \alpha_{12}e^{-i2s\pi/n}$$

$$l = 1, 2, \dots, m; \quad s = 0, 1, \dots, n-1.$$

Therefore if all the matrices $\{A_{s,m}, s = 0, 1, \dots, n-1\}$ are invertible, then so is the matrix $\varphi(B, W)$. It is easy to see that

$$\det(A_{s,m}) = \lambda_{ms} \det(A_{s,m-1}) - \theta_{m-1}^2 \det(A_{s,m-2}).$$

When the matrix $\varphi(B, W)$ is invertible, then the model (2.5) can be written as

$$\xi(t) = \varphi^{-1}(B, W)(I_n \otimes \Phi)\xi(t-1) + \varphi^{-1}(B, W)\epsilon(t). \quad (2.6)$$

Therefore $\{\xi(t)\}$ is a temporal vector AR(1) process. Define

$$\Upsilon = \varphi^{-1}(B, W)(I_n \otimes \Phi).$$

Then this vector AR(1) process is stationary if all the eigenvalues of Υ lie inside the unit circle and $\sigma_i^2(t) \equiv \sigma^2$.

In model (2.1), if $\alpha_{11} \equiv \alpha_1$, $\alpha_{12} \equiv \alpha_2$, and $\theta_l \equiv \theta$, i.e., the spatial parameters do not depend on latitude, we call the model a homogeneous space-time model. In this case, define

$$\lambda_s = 1 - \alpha_1 e^{i2s\pi/n} - \alpha_2 e^{-i2s\pi/n}$$

$$s = 0, 1, \dots, n-1$$

and

$$U_T = \begin{bmatrix} 0 & 0 & 0 & \dots & 0 & 0 \\ 1 & 0 & 0 & \dots & 0 & 0 \\ 0 & 1 & 0 & \dots & 0 & 0 \\ \dots & \dots & \dots & \dots & \dots & \dots \\ 0 & 0 & 0 & \dots & 1 & 0 \end{bmatrix}_{T \times T}$$

Then

$$A_{s,m} = \lambda_s I_m - \theta(U_m + U'_m).$$

The determinant of $A_{s,m}$ is

$$\det(A_{s,m}) = \begin{cases} \frac{(\lambda_s + \sqrt{\lambda_s^2 - 4\theta^2})^{m+1} - (\lambda_s - \sqrt{\lambda_s^2 - 4\theta^2})^{m+1}}{2^{m+1} \sqrt{\lambda_s^2 - 4\theta^2}} & \text{if } \lambda_s^2 \neq 4\theta^2 \\ (m+1)(\lambda_s/2)^m & \text{if } \lambda_s^2 = 4\theta^2 \end{cases} \quad (2.7)$$

and the determinant of $\varphi(B, W)$ is

$$\det(\varphi(B, W)) = \prod_{s=0}^{n-1} \det(A_{s,m}).$$

2.3 Likelihood function for the parameters in space-time model

In this section, we derive the likelihood function for the parameters in the space-time model (2.1). Define

$$\omega(1) = \varphi(B, W)\xi(1) - \epsilon(1).$$

Furthermore, assume that $\omega(1)$ is independent of $\{\epsilon(t), t = 1, 2, \dots, T\}$ and normally distributed with mean zero and covariance matrix Σ_0 . Let

$$\begin{aligned} \xi &= [\xi(1), \xi(2)', \dots, \xi(T)']', \\ \epsilon &= [\epsilon(1)', \omega(1)', \epsilon(2)', \dots, \epsilon(T)']', \\ \alpha &= [\alpha_{11}, \alpha_{21}, \dots, \alpha_{m1}]', \\ \theta &= [\theta_1, \theta_2, \dots, \theta_{m-1}]', \\ \phi &= [\phi_1, \phi_2, \dots, \phi_m]', \\ \Sigma(t) &= \text{Diag}(\sigma_i^2(t)) \end{aligned}$$

and

$$A = I_T \otimes \varphi(B, W), \quad D = U_T \otimes (I_n \otimes \Phi).$$

Then the model (2.5) can be written as

$$(A - D)\xi = \epsilon$$

Since the covariance matrix of $\epsilon(t)$ is

$$\text{Cov}(\epsilon(t), \epsilon'(t)) = I_n \otimes \Sigma(t),$$

the covariance matrix of ϵ is

$$\begin{aligned} \Sigma &= \text{Cov}(\epsilon, \epsilon') \\ &= \begin{bmatrix} \Sigma^*(1) & O & O & \dots & O \\ O & I_n \otimes \Sigma(2) & O & \dots & O \\ \dots & \dots & \dots & \dots & \dots \\ O & O & O & \dots & I_n \otimes \Sigma(T) \end{bmatrix} \end{aligned}$$

where $\Sigma^*(1) = I_n \otimes (\Sigma(1) + \Sigma_0)$. Therefore the determinant of Σ is

$$\det(\Sigma) = \det(\Sigma^*(1)) \prod_{l=1}^m \prod_{t=1}^T \sigma_l^{2n}(t).$$

Furthermore, when $\varphi(B, W)$ is invertible, we have

$$V = \text{Cov}(\xi, \xi') = (A - D)^{-1} \Sigma (A' - D')^{-1}$$

and

$$V^{-1} = (A - D)' \Sigma^{-1} (A - D).$$

Since

$$\begin{aligned} |(A - D)'(A - D)| &= |\varphi(B, W)|^{2T} \\ &= \left\{ \prod_{s=0}^{n-1} [\det(A_{s,m})]^2 \right\}^T. \end{aligned}$$

Hence

$$\begin{aligned} \det(V^{-1}) &= \{\det[\Sigma^*(1)]\}^{-1} \left\{ \prod_{l=1}^m \prod_{t=2}^T \sigma_l^{2n}(t) \right\} \\ &\times \left\{ \prod_{s=0}^{n-1} [\det(A_{s,m})]^2 \right\}^T. \end{aligned}$$

The log-likelihood function for the parameters in the model (2.1) is

$$\begin{aligned} l(\alpha, \theta, \phi, \Sigma|\xi) &= \\ &= -\frac{T}{2} \log(2\pi) + \frac{T}{2} \sum_{s=0}^{n-1} \log[\det(A_{s,m})]^2 \\ &- \frac{1}{2} \log\{\det[\Sigma^*(1)]\} - \frac{n}{2} \sum_{l=1}^m \sum_{t=2}^T \log \sigma_l^2(t) \\ &- \frac{1}{2} \xi' V^{-1} \xi. \end{aligned} \quad (2.8)$$

3 Applications of the space-time models

In this section, the space-time models are applied to the TOMS satellite column ozone data, which are over the period November 1978 to May 1990. Similar to the analysis performed by Niu (1991), we divide the globe into 10° latitude by 10° longitude blocks, and label the zonal range 0°N - 10°N as latitude 05N, 10°N - 20°N as latitude 15N, and so on. The monthly averages for total column ozone are formed for each geographic block. In each latitude zone, we have 36 longitudinal monthly average ozone series. For latitudes 75S, 85S, 75N and 85N,

there are some missing values since the satellite could not observe data over the polar region in the night. For this study, we consider only the ozone data in latitudes 65S-65N. Our main purpose here is to examine the statistical structure of the TOMS ozone data. The long-term trend analysis based on this data set will be discussed in another paper.

For the additive model (1.5), we model the seasonal component $S_{lj}(t)$ and the trend component $R_{lj}(t)$ as follows.

$$\begin{aligned} S_{lj}(t) &= \beta_{lj0} + \beta_{lj1} \sin(2\pi t/12) + \beta_{lj2} \cos(2\pi t/12) \\ &+ \beta_{lj3} \sin(4\pi t/12) + \beta_{lj4} \cos(4\pi t/12) \end{aligned}$$

and

$$R_{lj}(t) = \gamma_{lj} \frac{t}{12}, \quad t = 1, 2, \dots, T.$$

For the monthly average TOMS ozone data, we first fit the additive model (1.5) to the series in each block by the least squares method. Then using the de-seasonalized and de-trended series

$$\{\xi_{lj}(t), l = 1, 2, \dots, m; j = 1, 2, \dots, n; t = 1, 2, \dots, T\},$$

i.e., the residuals from the simple regression model (1.5), we calculate the sample temporal autocorrelations, the sample longitudinal correlations and the sample latitudinal correlations for the error terms $\{\xi_{lj}(t)\}$ in the northern hemisphere and the southern hemisphere, respectively. For a fixed time t , the longitudinal and latitudinal neighbor schemes of $\{\xi_{lj}(t), l = 1, 2, \dots, m; j = 1, 2, \dots, n\}$, are shown in Figure 1a and Figure 1b. Figure 2a and Figure 2b present the sample autocorrelations from order 1 to order 10, where the first order autocorrelations appear around 0.5. In fact, the sample autocorrelations of $\{\xi_{lj}(t)\}$ decrease toward the high latitudes. The largest first order autocorrelations occur for the series in latitudes 05N and 05S with values around 0.8. The sample longitudinal correlations for the series $\{\xi_{lj}(t)\}$ in the two hemispheres, also from order 1 to order 10, are shown in Figure 3a and Figure 3b, respectively. From the plots, we can see that the first order longitudinal correlations are around 0.9, and the order 10 longitudinal correlations are still very positive. For the sample latitudinal correlations in the two hemispheres, we calculate only from order 1 to order 5 and present them in Figure 4a and Figure 4b, respectively. The first and second order latitudinal correlations are all around 0.7.

Since the sample autocorrelations, the sample longitudinal and latitudinal correlations are all very positive, we consider the the following STAR(2,1,1) model for the error series $\{\xi_{lj}(t)\}$ in the northern hemisphere and in the southern hemisphere, separately.

Figure 1. Longitudinal and Latitudinal Neighbor Scheme

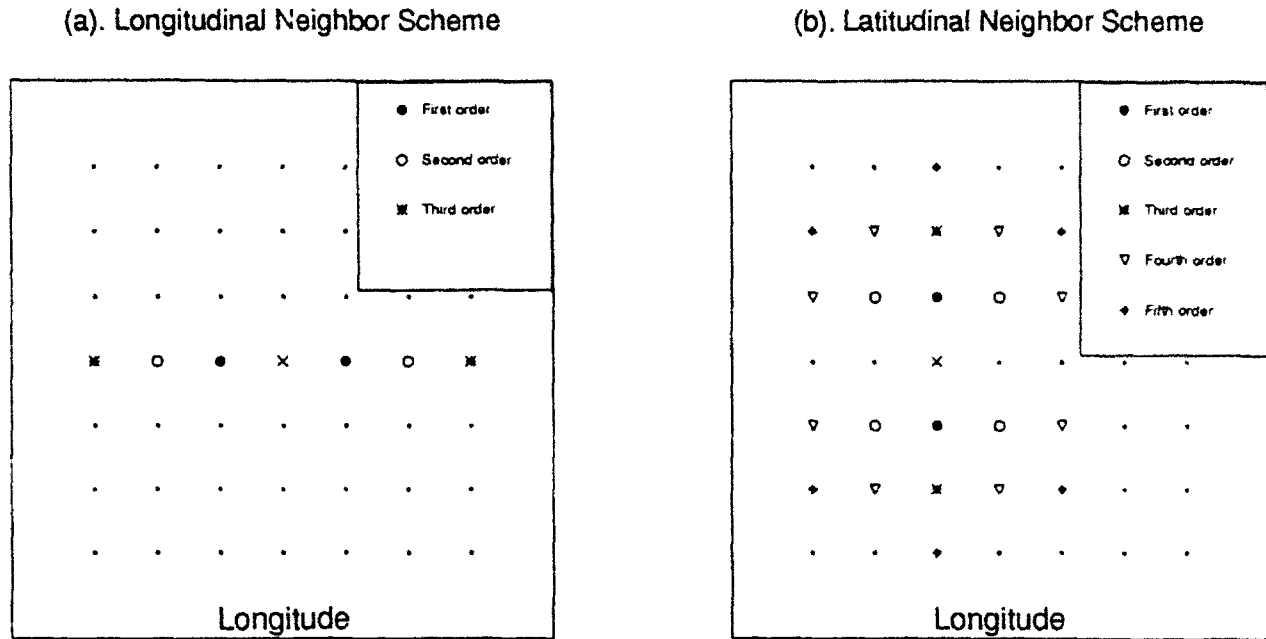


Figure 2. Sample Autocorrelations of the Residuals

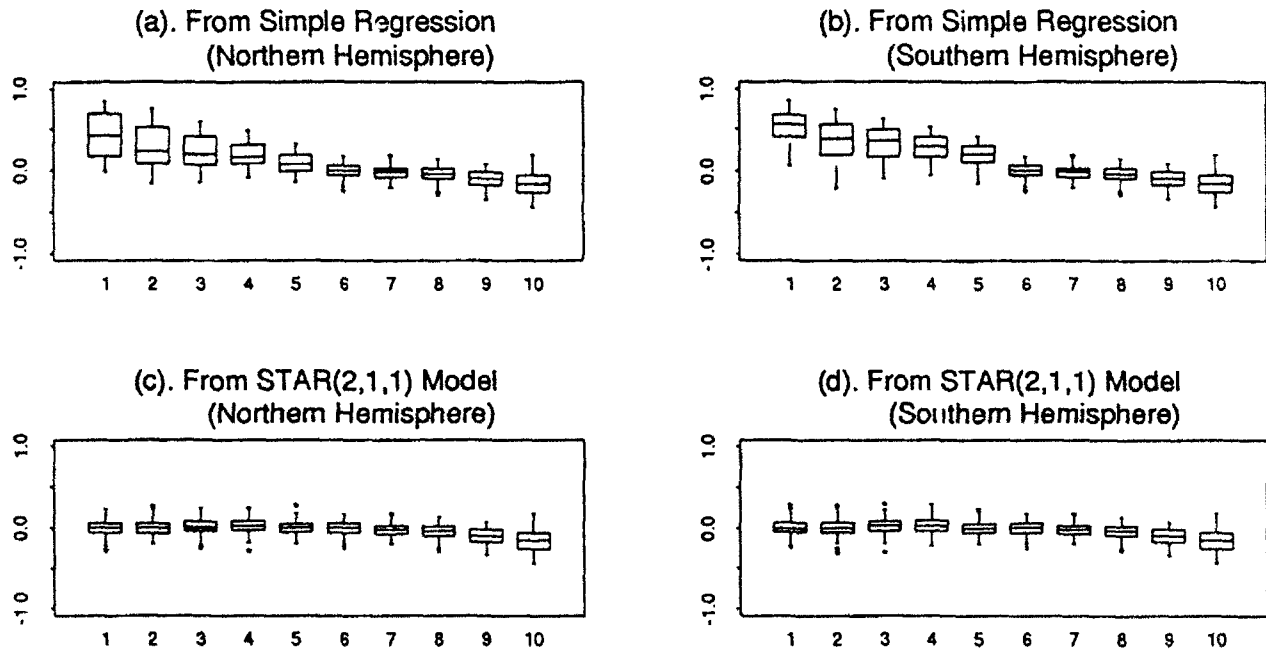


Figure 3. Sample Longitudinal Correlations of the Residuals

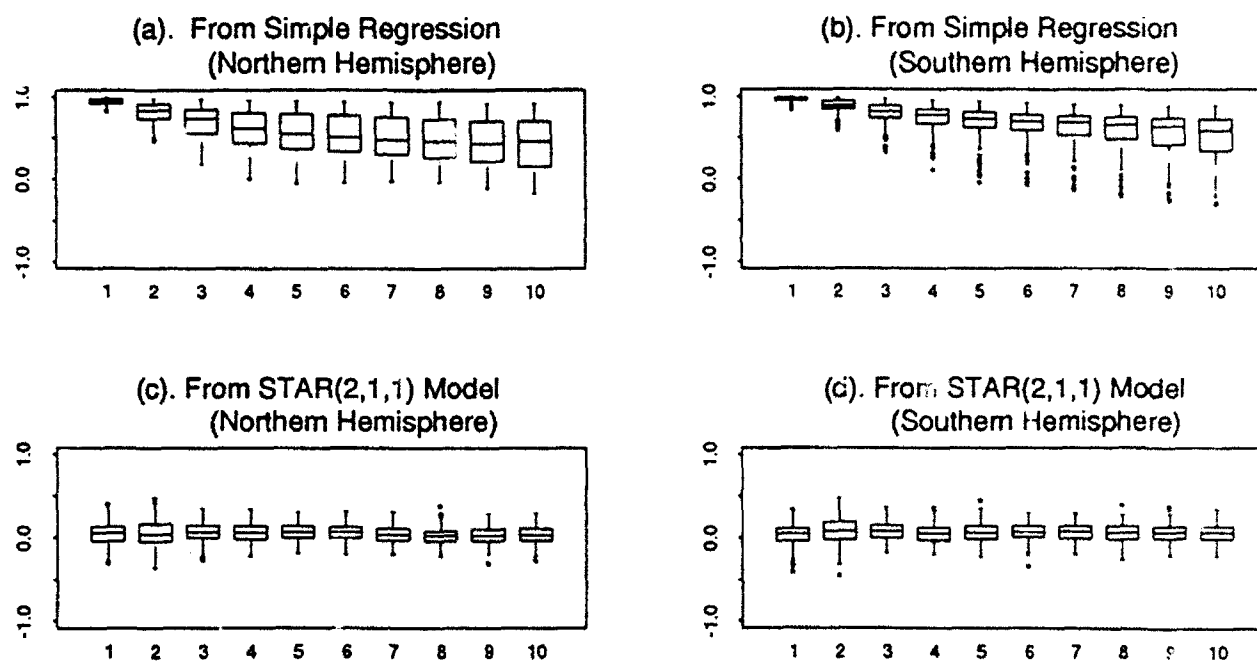


Figure 4. Sample Latitudinal Correlations of the Residuals

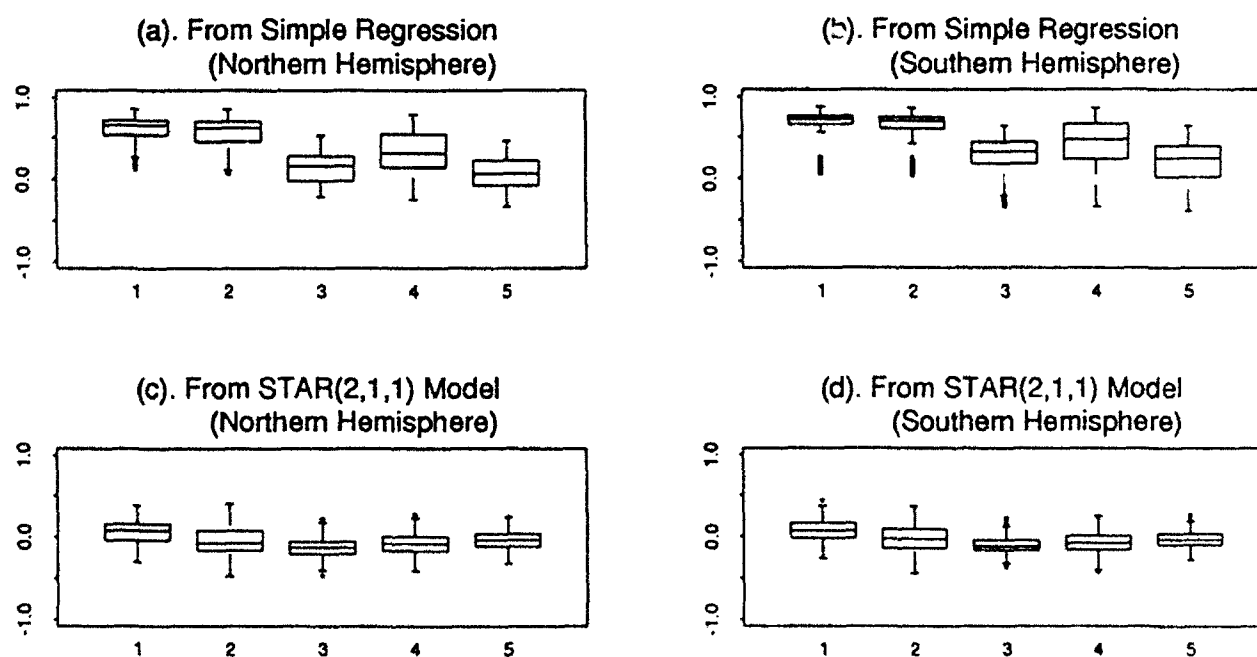


Table 1: Estimated Coefficients in the STAR(2,1,1) Model (Northern Hemisphere)

Lat.	Estimated Coefficients	
05N	$\alpha_{11} = 0.402$ (0.0048)	$\alpha_{12} = 0.065$ (0.0043)
	$\phi_1 = 0.055$ (0.0021)	$\theta_1 = 0.024$ (0.0010)
15N	$\alpha_{21} = 0.453$ (0.0043)	$\alpha_{22} = -0.012$ (0.0044)
	$\phi_2 = 0.031$ (0.0023)	$\theta_2 = 0.067$ (0.0016)
25N	$\alpha_{31} = 0.487$ (0.0037)	$\alpha_{32} = -0.067$ (0.0038)
	$\phi_3 = 0.023$ (0.0025)	$\theta_3 = 0.084$ (0.0017)
35N	$\alpha_{41} = 0.536$ (0.0028)	$\alpha_{42} = -0.124$ (0.0029)
	$\phi_4 = 0.013$ (0.0023)	$\theta_4 = 0.094$ (0.0016)
45N	$\alpha_{51} = 0.569$ (0.0022)	$\alpha_{52} = -0.148$ (0.0023)
	$\phi_5 = 0.002$ (0.0017)	$\theta_5 = 0.087$ (0.0013)
55N	$\alpha_{61} = 0.564$ (0.0022)	$\alpha_{62} = -0.129$ (0.0023)
	$\phi_6 = 0.001$ (0.0012)	$\theta_6 = 0.071$ (0.0011)
65N	$\alpha_{71} = 0.558$ (0.0024)	$\alpha_{72} = -0.084$ (0.0026)
	$\phi_7 = 0.001$ (0.0010)	

Table 2: Estimated Coefficients in the STAR(2,1,1) Model (Southern Hemisphere)

Lat.	Estimated Coefficients	
05S	$\alpha_{11} = 0.392$ (0.0049)	$\alpha_{12} = 0.068$ (0.0050)
	$\phi_1 = 0.070$ (0.0024)	$\theta_1 = 0.036$ (0.0012)
15S	$\alpha_{21} = 0.443$ (0.0044)	$\alpha_{22} = -0.016$ (0.0045)
	$\phi_2 = 0.033$ (0.0026)	$\theta_2 = 0.082$ (0.0018)
25S	$\alpha_{31} = 0.494$ (0.0036)	$\alpha_{32} = -0.079$ (0.0036)
	$\phi_3 = 0.023$ (0.0026)	$\theta_3 = 0.085$ (0.0016)
35S	$\alpha_{41} = 0.553$ (0.0025)	$\alpha_{42} = -0.132$ (0.0026)
	$\phi_4 = 0.007$ (0.0019)	$\theta_4 = 0.078$ (0.0013)
45S	$\alpha_{51} = 0.543$ (0.0024)	$\alpha_{52} = -0.107$ (0.0025)
	$\phi_5 = -0.003$ (0.0013)	$\theta_5 = 0.070$ (0.0012)
55S	$\alpha_{61} = 0.522$ (0.0028)	$\alpha_{62} = -0.060$ (0.0029)
	$\phi_6 = -0.006$ (0.0010)	$\theta_6 = 0.037$ (0.0010)
65S	$\alpha_{71} = 0.516$ (0.0031)	$\alpha_{72} = -0.029$ (0.0032)
	$\phi_7 = 0.003$ (0.0006)	

$$\begin{aligned} \xi_{lj}(t) = & \alpha_{11}[\xi_{l,j-1}(t) + \xi_{l,j+1}(t)] \\ & + \alpha_{12}[\xi_{l,j-2}(t) + \xi_{l,j+2}(t)] \\ & + \theta_{l-1}\xi_{l-1,j}(t) + \theta_l\xi_{l+1,j} \\ & + \phi_l\xi_{lj}(t-l) + \epsilon_{lj}(t) \end{aligned} \quad (3.1)$$

$$l = 1, 2, \dots, 7; \quad j = 1, 2, \dots, 36; \quad t = 1, 2, \dots, T.$$

In fact, we have tried to use a STAR(1,1,1) model to fit the de-seasonalized and de-trended series, but found the STAR(1,1,1) model could not remove the longitudinal correlations completely.

Using the maximum likelihood method, we estimate the parameters in the STAR(2,1,1) model for both hemispheres, and calculate the residual series $\{\epsilon_{lj}(t)\}$. The parameter estimates and their estimated standard errors, for the northern hemisphere and the southern hemisphere, are presented in Table 1 and Table 2, respectively. Except for the temporal coefficients in latitudes 45N-65N, all other parameter estimates are statistically significant. The longitudinal correlations for the de-seasonalized and de-trended series are much stronger than the latitudinal and temporal correlations.

Finally, the sample autocorrelations, the sample longitudinal and latitudinal correlations for the residual series $\{\hat{\epsilon}_{ij}(t)\}$ are calculated and presented in Figure 2c, Figure 2d, Figure 3c, Figure 3d, Figure 4c and Figure 4d, respectively. From these plots, we see that all the sample correlations are around zero.

4 Conclusions

Satellite temperature and ozone data have global coverage, which enable us to assess changes in temperature and ozone as a function of location. In this paper, we extend the space-time models for satellite ozone and temperature data introduced by Niu (1991). In the spatial direction, both the longitudinal and latitudinal interactions are considered. The space-time models are parsimonious and the parameters in the models are intuitively meaningful. After some appropriate transformations, the space-time models can be expressed as special vector AR processes. The covariance matrices of the error terms in the models have a block circular structure. The stationarity conditions for the error processes are studied, and the likelihood function for the parameters in the models are obtained. The space-time models are applied to the de-seasonalized and de-trended TOMS ozone data. After fitting a STAR(2,1,1) model to the de-seasonalized and de-trended series in the northern hemisphere and the southern hemisphere, respectively, diagnostic plots show that temporal, longitudinal and latitudinal correlations are all successfully removed.

References

- Besag, J. On the correlation structure of some two dimensional stationary processes, *Biometrika*, 59, 43-48, 1972.
- Good, I. J., On the inversion of circulant matrices, *Biometrika*, 42, 185-186, 1954.
- Niu, X., Space-Time ARMA Models for Satellite Ozone Data, Ph. D. Thesis, Department of Statistics, University of Chicago, 1991.
- Reinsel, G. C., G. C. Tiao, M. N. Wang, R. Lewis, and D. Nychka, Statistical Analysis of Stratospheric Ozone Data for the Detection of Trends, *Atmos. Environ.*, 15, 1569-1577, 1981.
- Reinsel, G. C., G. C. Tiao, S. K. Ahn, P. Marian, S. Basu, J. J. DeLuisi, C. L. Mateer, A. J. Miller, P. S. Connell, and D. J. Wuebbles, An Analysis of the 7-Year Record of SBUV Satellite Ozone Data: Global Profile Features and Trends in Total Ozone, *J. Geophys. Res.*, 93, 1689-1703, 1988.
- Tjøstheim, D., Statistical spatial series modeling, *Adv. Appl. Prob.*, 10, 130-154, 1978.
- Tjøstheim, D., Autoregressive modeling and spectral analysis of array data in the plane, *IEEE Transactions on Geoscience and Remote Sensing*, GE19, 15-24, 1981.
- Tjøstheim, D., Statistical spatial series modeling II: some further results on Unilateral Processes, *Adv. Appl. Prob.*, 15, 562-584, 1983.
- Whittle, P., On stationary processes in the plane, *biometrika*, 41, 434-449, 1954.

Space-time modeling of simply connected objects: An application to detection of left ventricular cardiac boundaries from ultrasound images

Geir Storvik*

Norwegian Computing Center,
Box 114 Blindern,
N-0314 Oslo, Norway

Paul Switzer

Dept. of Statistics
Stanford University
California 94305

Abstract

Simply connected objects varying over time often occur in medical imaging problems. A typical example is left ventricular cavity in ultrasound images. Much *a priori* information is available both about the shape and the motion in time of the objects. Construction of *a priori* models that take this information into consideration in a suitable form is a challenging problem.

In this paper we propose the use of a chain representation of nodes for describing the boundary of simply connected objects. Similar representations are widely used in the recognition part of image analysis (i.e. recognition of handwritten symbols). Use of the chain representation in the image *segmentation* step makes it possible to handle more global models than what is possible for Markov Random Field models without making any strong restrictions on the shape of the boundary. Furthermore, the timeaspect may be incorporated by matching successive boundaries in time. The approach is applied to the problem of detecting left ventricular cardiac boundaries from ultrasound images.

1. Introduction

The Bayesian approach has been widely used to incorporate prior knowledge into the analysis of images (see Besag [2]). Markov random fields (MRF) has been popular for describing both the prior and posterior distribution on the original images. The equivalence between MRF and Gibbs fields makes modeling with MRF easy, but the popularity of MRF models is also due to their generality and the existence of algorithms like simulated annealing (SA) (Geman and Geman [4]) for finding maximum a posteriori (MAP) estimates.

Although in principle any characteristic may be included in the MRF models, computational considerations make it necessary to restrict oneself to local char-

acteristics. This can lead to large-scale characteristics of the model that are undesirable.

In some problems of image analysis, *a priori* information is available on the content in the images. An example that will be used throughout the paper is the problem of automatic detection of left ventricular cavity boundaries from sequential ultrasound images. Information on shape, size and motion is available and should be taken into consideration.

There are several ways of building global information into a model. The obvious way to do so inside the MRF models is to increase the neighborhood system sufficiently, but as noted above this may not be practical.

Representation of objects in an image by a chain (or vector) of nodes that together represent the boundaries of the objects is commonly used in the recognition step of an image analysis. This is usually called a polygon-representation of an object. In this paper we will discuss the possibilities of using such a representation also in the boundary identification (or segmentation) step. We will show that such a representation makes the modeling easier, and computation easier.

The next section will present the example that will be used as an illustration throughout the paper. In section 3. we discuss the boundaryrepresentation of objects in an image. In section 4. we discuss how prior knowledge may be modeled. The computational and algorithmic aspects of the approach are discussed in section 5.. In section 6. we show results for boundarydetection of the left ventricular cavity. A summary is given in section 7..

2. Example

The example that will be used throughout the paper is a sequence of noisy ultrasound images containing left ventricular cavity. The aim with these images is to identify the boundaries of the ventricle. The images are taken so frequently in time that models containing both spatial and time-dynamic aspects should be considered.

*Supported by Royal Norwegian Council for Scientific and Industrial Research, Grant ST.10.12.220289.

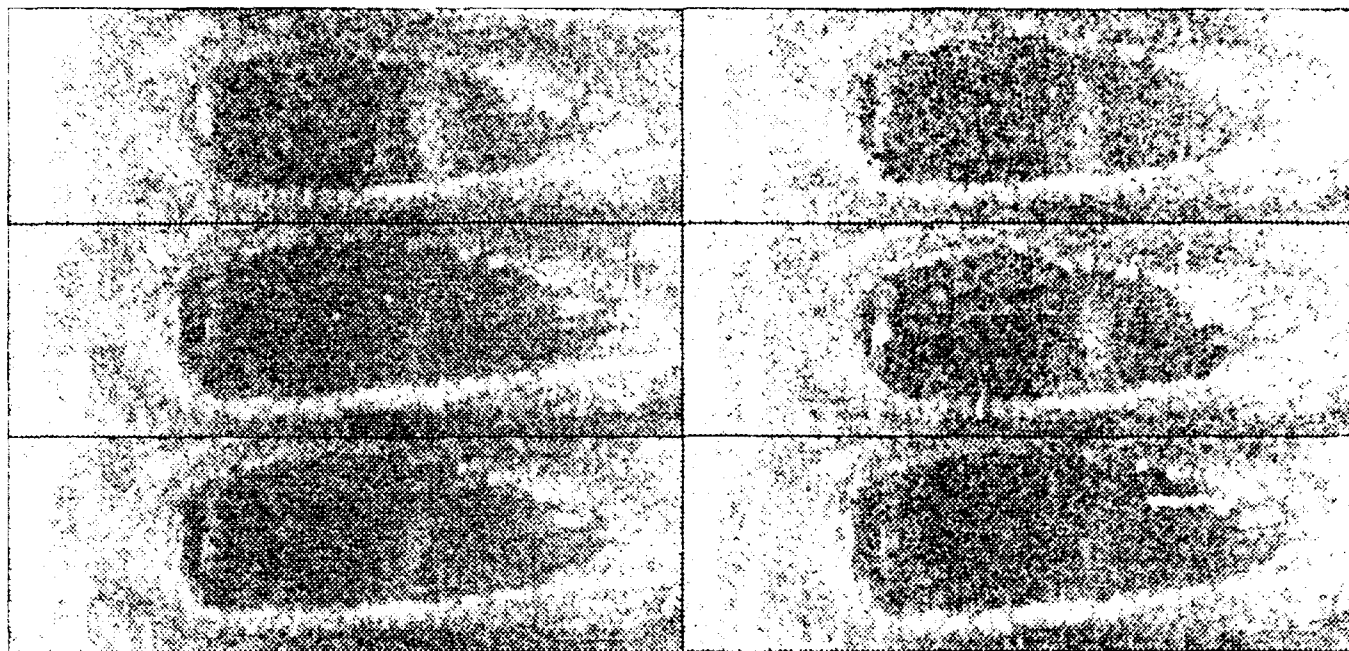


Figure 1. A sequence of 6 ultrasound images from left ventricular cavity.

Figure 1 shows a sequence of 6 ultrasound images containing the left ventricular cavity. The ventricle is shown as the darker area in the middle of the images. At the bottom of each image there is a thin dark area which corresponds to the right ventricle. Furthermore, there is a dark area at the top of the images corresponding to an artery going out from the left ventricle. Inside the ventricle there are some areas of muscles that are shown as lighter areas near the boundary on the left and a little bit to the right of the center. Automatic methods for boundary detection are usually confused by these areas.

Automatic detection of the ventricular cavity boundary has been an object of interest, both because it is an attractive clinical tool and because of the challenge in making use of the large amount of prior information in hand.

In Fredland and Adam [3] and Lundervold and Tuxen [4], an approach where the boundaries were represented as lengths from a pre-specified centerpoint along a number of radii was used. This representation makes it easy to incorporate both spatial and time dynamic aspects. The representations however the object to be star-shaped with respect to the centerpoint. Furthermore, since the distance between radii increase with increasing distance from the centerpoint, the resolution will vary along the boundary.

Dynamic (active) contours (Kass et al. [8], Amni et al. [1]) were applied in Olstad [10]. Fast algorithms using dynamic programming exist for this approach. How-

ever an initial boundary in the neighborhood of the final solution is required. Furthermore, extension to the time aspect is difficult to incorporate.

3. Boundary representation

A common assumption in image analysis is that a set of records $y = \{y_k : k = 1, \dots, T\}$ is generated by a (stochastic) degradation D of the true pixel image $x = \{x_k : k = 1, \dots, S\}$ (Bevilacqua [2]). The two sets T and S are dependent on the type of problem. Generally, T is determined by the sensing device, while S can be controlled by the user. In the literature, S is usually chosen to be deterministic, and either identical or closely related to T . In many cases, there may be preferable to use a coarser or finer representation for S , or even vary the scale during processing (Gidas [5]). In this paper, however, we will discuss a choice of S that is more loosely related to T , but that is more closely related to contents of x .

Under the assumption that the image consists of only one object that has a simply connected domain, the image x is completely defined by the boundary of the object.

Polygon-representation of an object is a representation where the boundary of the object is defined by a vector of nodes giving coordinates of points of lines in a circular (clockwise) manner. Between each point, the boundary is defined by a straight line. Representation of objects

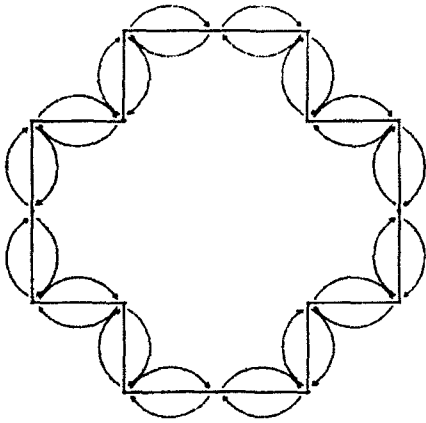


Figure 2: Chain-representation of the boundary for a simply connected object.

in this manner has been considered widely in the object recognition literature (Persoon and Fu [11], Zahn and Roskies [13]).

Use of such representations in the *segmentation* step of image analysis appears in the recent literature. For a fixed number of nodes, Grenander and Keenan [6] describes a global, pattern-theoretic model of shape. Furthermore, in the active contour approach (Kass et.al [8], Amini et.al [1]) a similar representation is used although the shape modeling is quite different.

Fixing the number of nodes on the boundary limits the set of possible configurations. To utilize the resolution of the observed image, we would like the boundary to follow the edges of the pixels. In order to do so, we will use a representation where the number of nodes are stochastic. In this representation each node contain, in addition to its localization, two pointers, one to the previous node, another to the succeeding node. Figure 2 illustrate this representation, which will be called the *chain-representation*. The nodes will be assumed to be located at the corner of the pixels with distance between succeeding nodes being equal to the length of a pixel-edge. In section 5. we will see that this representation also makes it easy to calculate shape characteristics and perform changes on the boundary.

4. Models

Energy-functions are usually easier to formalize than probabilities. We will therefore assume that the *a priori* model is of the form

$$\pi(x) = \frac{1}{Z} e^{-\beta' U(x)/T}, \quad (4.1)$$

where $U(\cdot) = \{U_1(\cdot), \dots, U_p(\cdot)\}$ is called the energy function, and T is the "temperature". β is a vector containing the parameters in the model while Z is a normalizing constant, usually unknown because of the huge number of possible configurations x .

Most of the energy functions we will considered may be written as

$$U_q(x) = \frac{1}{|x|} \sum_{i \in S} V_q(x; i) \quad (4.2)$$

where $V_q(x; i)$ is a *potential* assigned to node x_i (that could however depend on other nodes than node x_i) and where $|x|$ is the number of nodes in x . Breaking the energy-function down to potentials makes the modeling step easier. Written in this form, the models look similar to the Gibbs distribution. Note however that the number of nodes $|x|$ is stochastic.

A priori models that have been used for incorporating spatial aspects have mainly been concerned about some kinds of local smoothness. However, smoothness may be scale-dependent. Figure 3 shows two curves where the second one is the one that is most smooth at small scale, but the first one may in many cases fit better to our *a priori* assumption of the shape of the curve. This simple example shows that use of smoothness at different scales may be important.

For short-scale smoothness, a natural measure to use is the curvature of the boundary. Since we are assuming the boundary to follow the pixel-edges, the curvature may be defined by the angles at the node-points. For square pixels, only two angles are possible, 90° or 180° . The potential describing the small-scale smoothness is then defined to be

$$V(x; i) = \alpha_{\theta_i}, \quad (4.3)$$

where θ_i is the angle at node x_i while the α 's are parameters giving the weights for the different angles. Figure 4 shows a simulation from a model using this potential function. We see that the boundary tends to consist of straight lines in the same direction as the edges of the pixels. Furthermore it consists of many thin or narrow tails.

For larger scale smoothness the measure will depend on several nodes. In particular, it will depend on $x_{i-m}, \dots, x_i, \dots, x_{i+m}$ where x_{i-k} is defined to be the k th node before node x_i and similarly for x_{i+k} . Note that m in this case will be a parameter of the model. We will propose different measures for large scale aspects, although better choices exist.

Boundaries consisting of extended straight lines may, in some cases, be desirable. A measure for straight lines



Figure 3: Curves with different smoothness and convexity.

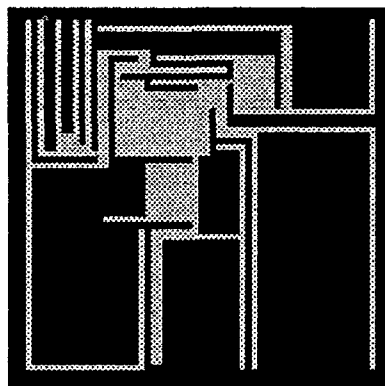


Figure 4: Simulation from an a priori model with small-scale smoothness (equation (4.3)).

that is directional independent is the correlation coefficient, which is defined by

$$V(x; i) = \frac{1}{2m} \sum_{i=-m}^m (x_i^x - \bar{x}^x)(x_i^y - \bar{x}^y) \quad (4.4)$$

where superscript x and y indicate the x - and y - coordinates of a node and \bar{x} is the average value of the nodes. Figure 5 shows simulations from this model. In this case we see that straight lines occur in any direction.

Smoothness at larger scales could be defined through Fourier descriptors calculated from $x_{i-m}, \dots, x_i, \dots, x_{i+m}$. Coefficients corresponding to different frequencies characterize curves on different scales. Large scale frequencies may then be used to model large scale characteristics. We refer to Storvik [12] for further details on this measure. Figure 6 shows a simulation based on this model. In this case more complicated structures occur with smoothness perhaps more appealing.

All the above models produce images with narrow features. The next measure we will propose is motivated by avoidance of narrow features. A property of narrow features is that nodes having a large distance between them on the boundary can be near each other in the euclidian

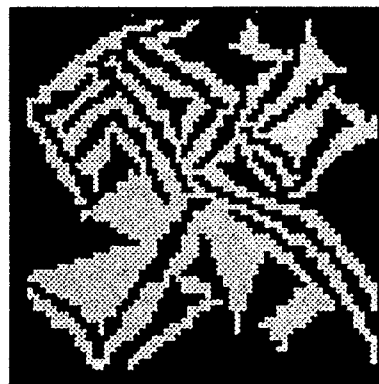


Figure 5: Simulation from an a priori model based on potentials (4.4).

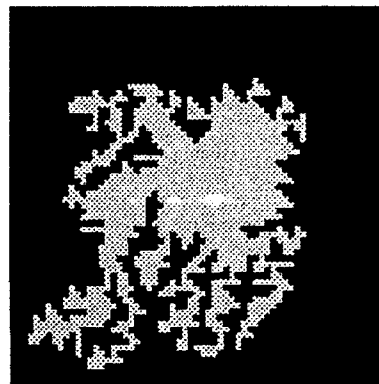


Figure 6: Simulation from an a priori model based on Fourier descriptors.

space. A measure for this could be

$$V(x; i) = \frac{d_a(x_{i-m}, x_{i+m})}{d_e(x_{i-m}, x_{i+m})}, \quad (4.5)$$

where d_a is a measure of the distance between two nodes on the boundary (i.e the arc length) while d_e is a measure of the distance between two nodes in the euclidian space. Figure 7 shows a simulation from this model.

The algorithms used for simulation and restoration using the chain-representation is described in section 5. A crucial aspect in these algorithms is however that the ratio $\pi(x')/\pi(x)$ is possible to calculate in reasonable time for configurations x and x' only differing at a small part of the boundary. So far the energy-functions we have considered have been built up by local measures, although on a larger scale than usually used. This makes the above ratio easy to calculate. Global measures could however be considered if either $\pi(x)$ or $\pi(x')\pi(x)$ is easily calculated. A measure falling in this class is one that compares the length of the boundary with the volume of

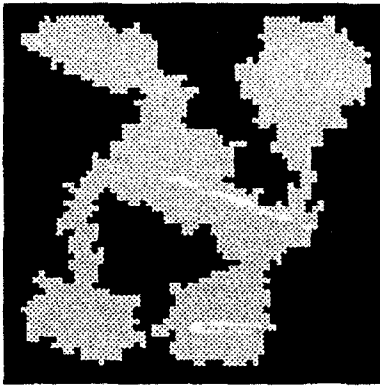


Figure 7: Simulation from an *a priori* model avoiding narrow tails.

the object:

$$U(x) = \frac{(\text{length of boundary})^\lambda}{\text{Volume of object}}. \quad (4.6)$$

For $\lambda = 2$, this energy-function is minimized for the object being a circle. Figure 8 shows a simulation from this model with $\lambda = 2$. We see that in this case the shape of the object is much more appealing globally.

We will now turn to the time-dynamic aspects. Consecutive images containing the same object taken over time, should indicate some smoothness of motion. First of all, since it is the same object that is observed, one would expect it to look quite similar in all the images. Secondly, assuming the curve is moving continuously over time, small intervals between recording of the images should indicate small changes in the curve. For the time-dynamic models that will be considered, we will assume that no translation or rotation has occurred between the recordings. The motion is assumed to be increase/decrease of the object and/or motion of small parts of the curve.

The time-dynamic models that will be considered, will be based on potential functions depending on the distance from points at one boundary to another boundary. Define

$$d(x_i(t); t') \quad (4.7)$$

as the distance from node x_i on the boundary at time t to the boundary at time t' . A simple model could be

$$V(x_i) = d(x_i(t); t-1) + d(x_i(t); t+1). \quad (4.8)$$

This potential will try to minimize the distance between succeeding boundaries.

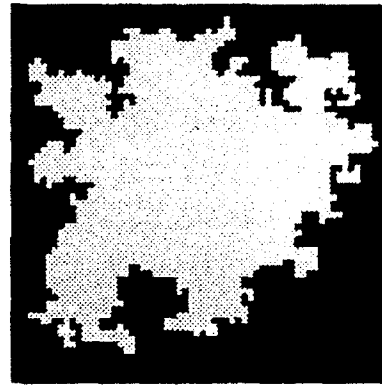


Figure 8: Simulation from the *a priori* model comparing length of boundary with volume.

A variant of model (4.8) that gave good results in the experiments was

$$V(x_i) = \max(0, d(x_i(t); t-1) - K) + \max(0, d(x_i(t); t+1) - K) \quad (4.9)$$

for some value of K . Giving large weights to this potential, this model essentially gives the restriction of the distance between the boundaries at each point to be less than K .

Some motion is however known to be present. Another model could therefore be that motion should be smooth. In that case the potential could be specified by

$$V(x_i) = ||d(x_i(t); t-1) - \frac{1}{2}[d(x_{i-1}(t); t-1) + d(x_{i+1}(t); t-1)]|| + ||d(x_i(t); t+1) - \frac{1}{2}[d(x_{i-1}(t); t+1) + d(x_{i+1}(t); t+1)]||. \quad (4.10)$$

where x_{i-1} and x_{i+1} are the previous and succeeding node of x_i , respectively.

A further possibility could be that motion should be at an approximately constant speed. This could be modeled by

$$V(x_i) = ||d(x_i(t); t-1) - d(x_i(t); t+1)||. \quad (4.11)$$

There are however some problems concerning this approach. First of all it is not obvious how to define the

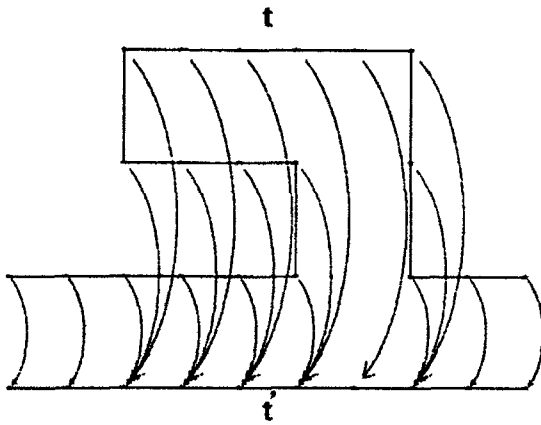


Figure 9: Example of matching nodes defined by minimization of distance without restrictions.

distance $d(x_i(t); t')$. Secondly, if the distance is properly defined, it has to be easy to calculate in order to be practical to use in the iterative segmentation algorithm.

We have chosen an approach where each node on the boundary have a corresponding or matching node on the other boundary. Assume $x_i(t), i \in S_t$ are the nodes on boundary t . Let $m_i(t'; t)$ be the index for the matching node on boundary t' . Then the distance could be defined by

$$d(x_i(t); t') = \|x_i(t) - m_i(t'; t)\|. \quad (4.12)$$

There is still however the problem of how $m_i(t'; t)$ should be defined. One choice could be to define it as the node on boundary t' which minimize (4.12). There are however some drawbacks with this definition as illustrated in Figure 9. We see that if $x_j(t)$ is a node between $x_i(t)$ and $x_k(t)$ on boundary t , this will not necessary imply that $m_j(t'; t)$ is a node between $m_i(t'; t)$ and $m_k(t'; t)$ on boundary t' . Such a feature would however be natural assumption. We will therefore define matching nodes as follows:

$\{m_i(t'; t), i \in S_t\}$ are defined to be the matching nodes on boundary t' corresponding to the nodes on boundary t if $\{m_i(t'; t), i \in S_t\}$ minimize

$$\sum_{i \in S_t} \|x_i(t) - m_i(t'; t)\| \quad (4.13)$$

subject to

$$m_{i-1}(t'; t) \leq m_i(t'; t) \leq m_{i+1}(t'; t). \quad (4.14)$$

where \leq in this context means before or equal to on the boundary.

The matching nodes then has to be found by an appropriate optimization algorithm. Such an algorithm is described in Storvik [12].

5. Computational and algorithmic aspects

Given the model and the data, simulating from the posterior distribution or finding the Maximum A Posterior (MAP) solution is usually desirable. The huge number of possible configurations makes it however impossible to search through all possibilities for finding the MAP solution, while the constant Z makes it difficult to simulate from the distribution. For MRF-models, iterative algorithms based on the Gibbs sampler or the Metropolis approach (Geman and Geman [4]) has been used for solving this problem. The use of these algorithms depend much on the simplicity of calculating the ratios $\pi(x')/\pi(x)$ or similarly $U(x') - U(x)$ when x' is similar to x .

We will follow a similar approach by constructing an algorithm based on simulated annealing where at each step, x may be changed to an x' which is "similar" to x . For MRF-models, x' being similar to x meant that x' and x only differed in one (or a few) pixels. In our case, we will define x' and x to be similar if they only differ in a small part of the boundary. Note that by only consider such changes, changes of pixels giving configurations that are not simply connected, will not be considered. This makes the algorithm much more efficient than algorithms searching through all pixels in the image.

Using energy-functions built up by locally dependent potentials, the potentials in the two configurations will be equal for most of the nodes. Since by (4.2), the energy-function is the average of the potentials and the number of nodes are stochastic, the feature of MRF's that most of the potentials will vanish in the difference $U(x') - U(x)$ will not apply in our case. However, utilizing recursive formula's for calculating means, $U(x') - U(x)$ is easily computed.

Simulation or optimization is then performed by using a Markov chain having as its stationary distribution the desired solution. The specification of the transition matrix P for the Markov chain is done by first specifying a transition matrix Q which gives the possible transitions from a configuration x . We have chosen Q such that a transition from x to x' is possible if the boundary is moved such that a limited number of pixels have changed from being inside to outside of the boundary or vice versa. Furthermore, the pixels that have changed are assumed to be connected.

If the position on the boundary at which the change can occur is selected at random, use of transition-probabilities as suggested in Hastings [7] for simulation result in that the reversibility condition $\pi(x)P_{x,x'} = \pi(x')P_{x',x}$ holds, implying convergence is ensured. Fur-

thermore, simulated annealing techniques can be used for optimizations.

Similar to the visiting-scheme used by Geman and Geman [4], a more appealing scheme would be where the changes on the boundary is done in a cyclic order along the boundary. In this case, the time-reversibility condition is no longer fulfilled, however convergence is possible to show also for this approach, see Storrer [12].

The time-dynamic models considered in section 4. creates a problem. The matching nodes are found by minimizing a global measure. When a boundary is changed, new matching nodes should therefore in principle be calculated. Such an approach would however be too time-consuming. We have therefore applied an approximative algorithm where for each change matching nodes are updated only for the nodes that are affected by the change. A global updating of the matching nodes are then done after every, say, N iterations. Of course, using such an approximative algorithm, convergence is no longer guaranteed, but experiments have shown the approach to work quite satisfactory.

6. Results

We will show results for restoration of the images given in Figure 1. All the energy-functions described in section 4. were tried.

For all the results shown, a simple energy-function for the data was used:

$$y_i \sim \begin{cases} N(\mu_i, \sigma_i) & \text{for pixel } i \text{ inside boundary} \\ N(\mu_o, \sigma_o) & \text{for pixel } i \text{ outside boundary} \end{cases} \quad (6.1)$$

The parameters $\mu_i, \sigma_i, \mu_o, \sigma_o$ were estimated from another image from the same sequence as those shown in Figure 1.

The model incorporating small scale smoothness is the most simple and requires least computation time. Figure 10 shows the best result from a selection of different parameter sets using the small scale smoothness model. Only the restoration of the first image in the sequence given in Figure 1 is given. The boundary has characteristics much like the one given in Figure 4 with many thin features. The result is far from satisfactory and large scale models were therefore necessary.

Among the more large scale models, the model avoiding narrow features performed best. Figure 11 shows the result for this model with $m = 15$. Large improvements are made compared to the result in Figure 10, but the boundaries still have some unappealing characteristics at the top of the images. Furthermore, the first images have some problems with the muscle at the lower right

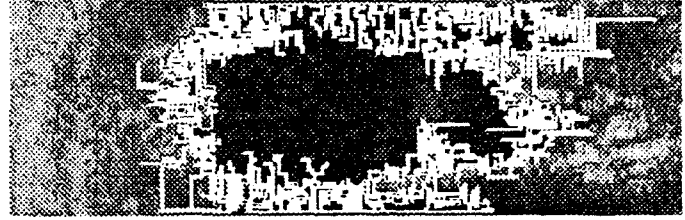


Figure 10: Restoration of first image using an a priori model with small scale smoothness.

part of the left ventricle and also with the right ventricle. All the images have problems with the artery at the top of the images, which is not surprising since the boundary between the left ventricle and the artery is almost invisible.

The use of the global measure comparing boundary length and the volume with $\lambda = 3$ (equation (4.6)) gave the result shown in Figure 12. We see that the restorations are quite similar to the ones given in Figure 11.

Finally, Figure 13 shows the restoration using a model incorporating both the spatial and temporal aspects. For the spatial aspect, the model comparing length of boundary with volume (equation (4.6)) was used, while for the temporal aspect, the potential given in (4.9) with $K = 3$ was chosen. These two energyfunctions were combined linearly (see equation (4.2)). A much better result is obtained with no confusion of the inside muscles or from the right ventricle. Some confusion is still present at the upper part of the images due to the artery but is much less than that for the restorations based only on spatial aspects.

7. Summary and discussion

We introduced an approach to boundary detection of images consisting of simply connected objects. The approach is based on a polygon representation of the object, which makes it possible to model aspects on a larger scale.

By defining distance between boundaries at succeeding images through matching nodes, the temporal aspect was also incorporated into the approach. This approach could also be used for images recorded at the same time but at different levels (i.e. MR-images).

Simulation experiments and restoration of ultrasound images containing left ventricular cavity show promising results. Further work should however be done on finding suitable energy-functions. The algorithm we used considers only small changes at each iteration step. Algorithms allowing larger changes are expected to converge faster.

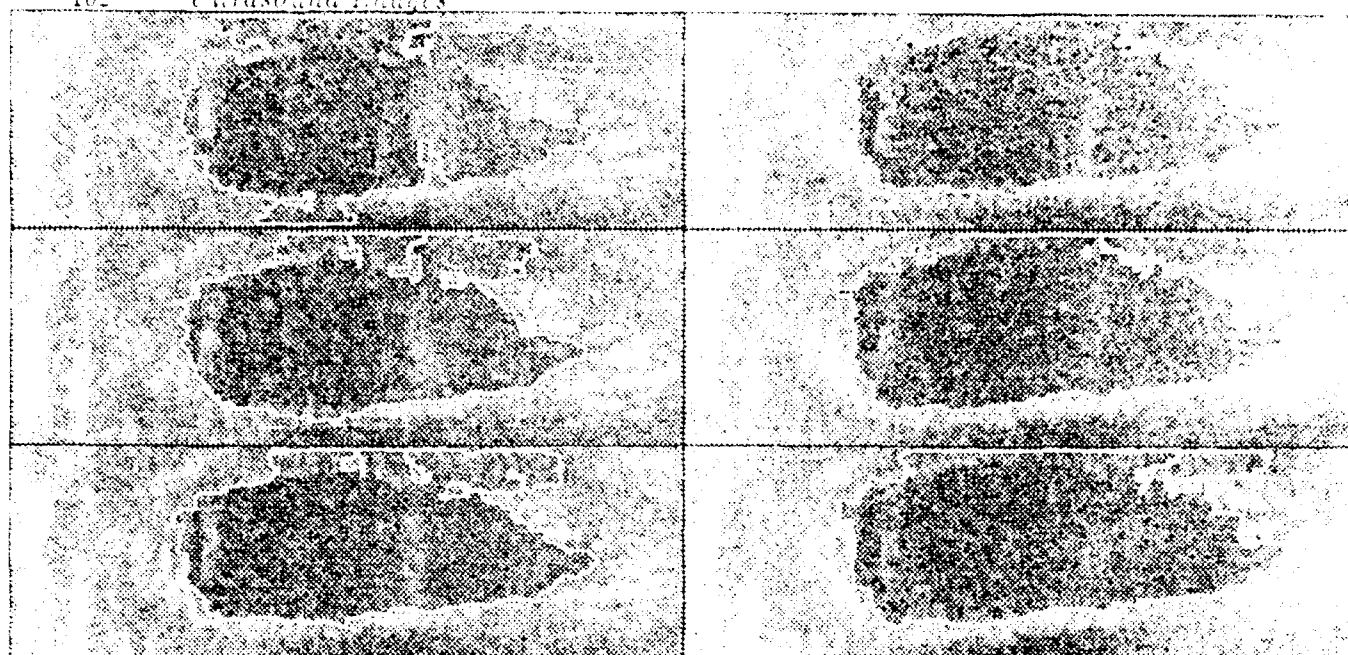


Figure 1. Fetal head in cross-section, 1988.

The same fetal head is shown in the same position, but with a different ultrasound image, showing a different view of the same head.

References

1. A. A. Averbach, I. J. W. Averbach, and J. A. Averbach, "The use of ultrasound in the diagnosis of fetal head malformations," *Journal of the American Medical Association*, vol. 261, no. 1, pp. 1-10, 1988.
2. J. B. Averbach, "The use of ultrasound in the diagnosis of fetal head malformations," *Journal of the American Medical Association*, vol. 261, no. 1, pp. 1-10, 1988.
3. J. B. Averbach, "The use of ultrasound in the diagnosis of fetal head malformations," *Journal of the American Medical Association*, vol. 261, no. 1, pp. 1-10, 1988.
4. J. B. Averbach, "The use of ultrasound in the diagnosis of fetal head malformations," *Journal of the American Medical Association*, vol. 261, no. 1, pp. 1-10, 1988.
5. J. B. Averbach, "The use of ultrasound in the diagnosis of fetal head malformations," *Journal of the American Medical Association*, vol. 261, no. 1, pp. 1-10, 1988.
6. J. B. Averbach, "The use of ultrasound in the diagnosis of fetal head malformations," *Journal of the American Medical Association*, vol. 261, no. 1, pp. 1-10, 1988.
7. J. B. Averbach, "The use of ultrasound in the diagnosis of fetal head malformations," *Journal of the American Medical Association*, vol. 261, no. 1, pp. 1-10, 1988.
8. J. B. Averbach, "The use of ultrasound in the diagnosis of fetal head malformations," *Journal of the American Medical Association*, vol. 261, no. 1, pp. 1-10, 1988.
9. J. B. Averbach, "The use of ultrasound in the diagnosis of fetal head malformations," *Journal of the American Medical Association*, vol. 261, no. 1, pp. 1-10, 1988.
10. J. B. Averbach, "The use of ultrasound in the diagnosis of fetal head malformations," *Journal of the American Medical Association*, vol. 261, no. 1, pp. 1-10, 1988.
11. J. B. Averbach, "The use of ultrasound in the diagnosis of fetal head malformations," *Journal of the American Medical Association*, vol. 261, no. 1, pp. 1-10, 1988.
12. J. B. Averbach, "The use of ultrasound in the diagnosis of fetal head malformations," *Journal of the American Medical Association*, vol. 261, no. 1, pp. 1-10, 1988.
13. J. B. Averbach, "The use of ultrasound in the diagnosis of fetal head malformations," *Journal of the American Medical Association*, vol. 261, no. 1, pp. 1-10, 1988.
14. J. B. Averbach, "The use of ultrasound in the diagnosis of fetal head malformations," *Journal of the American Medical Association*, vol. 261, no. 1, pp. 1-10, 1988.
15. J. B. Averbach, "The use of ultrasound in the diagnosis of fetal head malformations," *Journal of the American Medical Association*, vol. 261, no. 1, pp. 1-10, 1988.
16. J. B. Averbach, "The use of ultrasound in the diagnosis of fetal head malformations," *Journal of the American Medical Association*, vol. 261, no. 1, pp. 1-10, 1988.
17. J. B. Averbach, "The use of ultrasound in the diagnosis of fetal head malformations," *Journal of the American Medical Association*, vol. 261, no. 1, pp. 1-10, 1988.
18. J. B. Averbach, "The use of ultrasound in the diagnosis of fetal head malformations," *Journal of the American Medical Association*, vol. 261, no. 1, pp. 1-10, 1988.
19. J. B. Averbach, "The use of ultrasound in the diagnosis of fetal head malformations," *Journal of the American Medical Association*, vol. 261, no. 1, pp. 1-10, 1988.
20. J. B. Averbach, "The use of ultrasound in the diagnosis of fetal head malformations," *Journal of the American Medical Association*, vol. 261, no. 1, pp. 1-10, 1988.

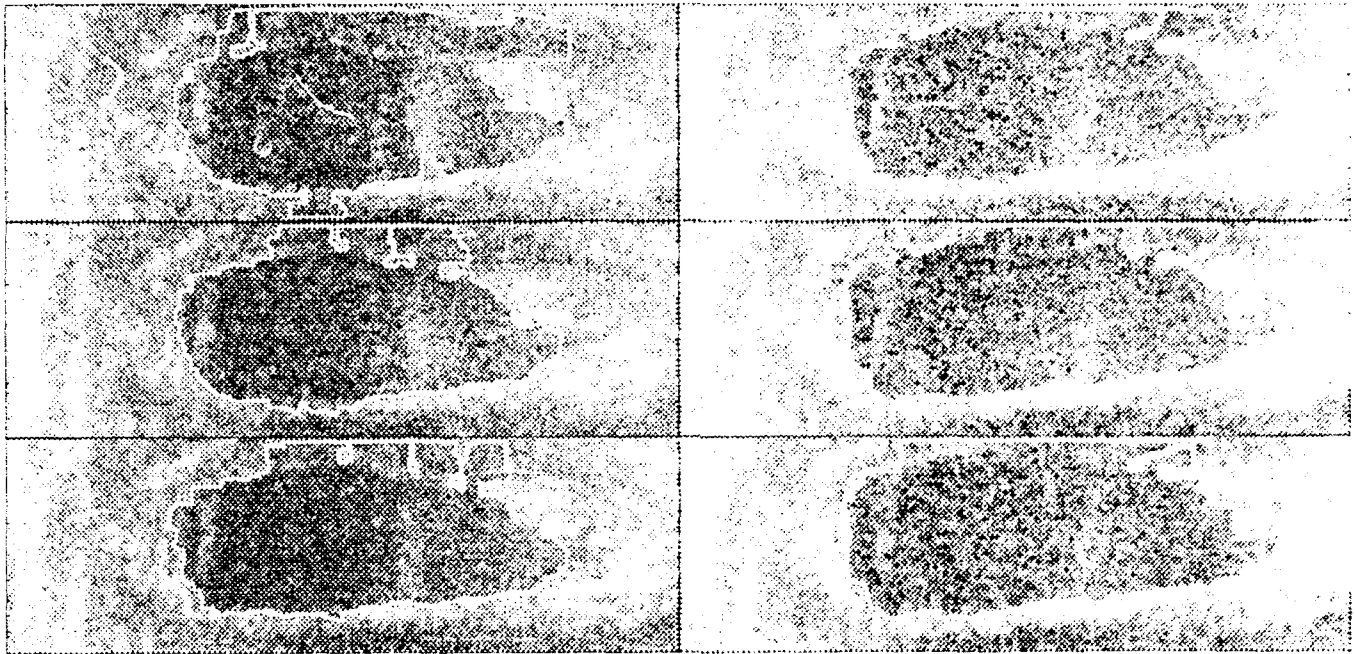


Fig. 1. Dorsal view of the larva of the caddisfly genus *Pseudostenophlebia* (left column) and *Pseudostenophlebia* (right column).

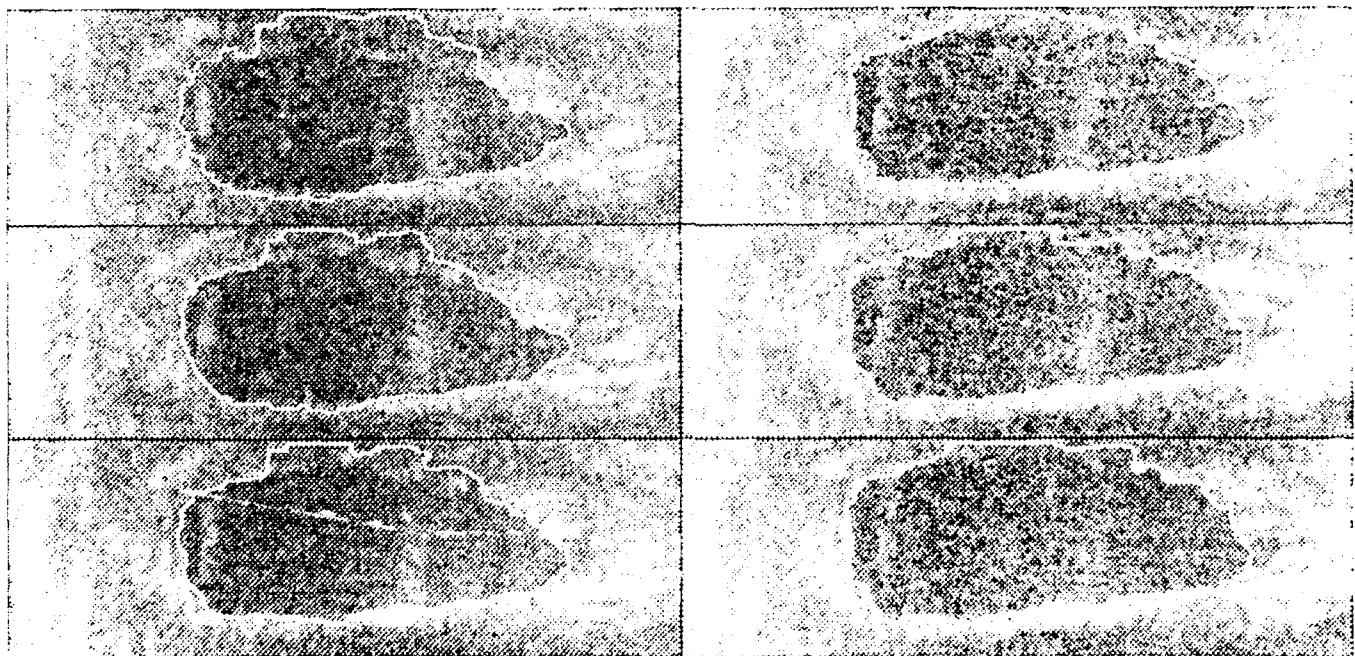


Fig. 2. Ventral view of the larva of the caddisfly genus *Pseudostenophlebia* (left column) and *Pseudostenophlebia* (right column).

Tools for managing the S object

O. Cherkaoui* and R. Ferland

Département de mathématiques et d'informatique,
Université du Québec à Montréal

Abstract

Like most object-oriented frameworks, New S allows: (1) a natural decomposition of applications, (2) software modularity, (3) easy reuse of software components written for previous applications. One of the advantages that the object-oriented framework of S offers is a high portability of code and reusability of object in other environments. To help the task of users in the development of systems, we have built a set of tools for managing S objects. From a basic object, we construct a hierarchical library of others objects. In order to manage any object, we have to deal with its library. The tools for managing the S objects allow portability between systems and the creation and archiving of object libraries.

1 Introduction

In the recent years, new software systems have been developed on a new programming paradigm: the object-oriented approach. S (Becker, Chambers and Wilks, 1988) was the first interactive program for data manipulation and display and for performance of a number of statistical calculations which use the object-oriented approach. Lisp-Stat system (Tierney, 1990) is also a new effective platform for a large number of statistical computing tasks and is strongly influenced by S.

The object-oriented programming promises far-reaching improvements in how people design, develop, and maintain software by offering long-term solutions to the problem of lack of portability and reusability. In an object-oriented environment, like New S, objects are generated by the users. In accordance to the reusability and the portability, we will address the questions:

- * How to manage an object from the developers to users?
- * How to make a library from an object?

This paper is organized as follows. In section 2, the motivation of the work is illustrated by two examples of practical situation in a programming

environment. The paradigm of the object-oriented programming is presented in the section 3. Two main advantages of the object-oriented programming, the reusability and the persistence are illustrated for the S environment. Section 4 describes the tools that have been developed to manage the S objects. Section 5 presents the conclusion and discussion on further work.

2 Motivation

We start with the justification and the motivation for this work.

2.1 First motivation

The S approach is useful for organizing functions and data objects by projects. But this approach takes less organization on the part of the user and it can waste space by saving unneeded objects. For each project, the designer builds a lot of objects. Most of the time, he has to reuse part of an object which he already developed for other projects. One solution is to create a library of objects. It is important to create a coherent library for a designer to share objects between projects.

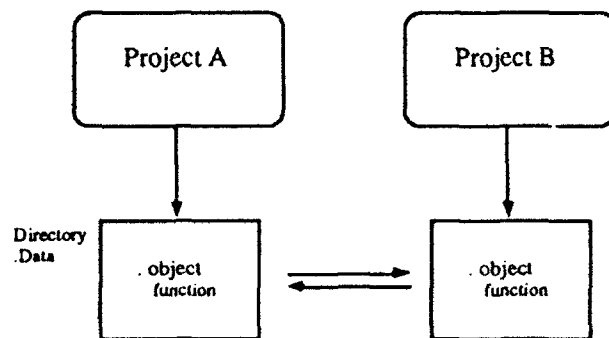


Fig. 1 : Managing object by project

2.2 Second motivation

The designer builds a model and generates a library of functions S. When creating its own application, a user bases his application-oriented model on the general model and reuses objects from the library already created by the designer. It is

* Work supported by NSREC Grant

important to export a library of objects from the designer to the user.

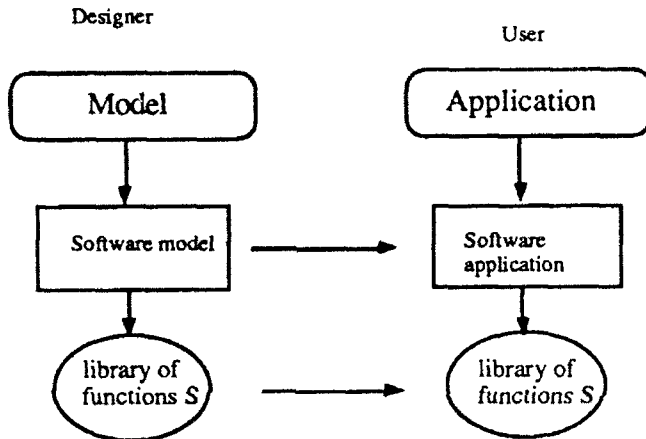


Fig. 2 : Portability between the designer and the user

3 Object-Oriented Programming

Object-oriented programming is often referred to a new programming paradigm (Budd, 1991). Other programming paradigms sometimes mentioned include: logical programming (Prolog) or functional programming. Object-oriented programming views a program as a collection of largely autonomous agents called *objects*. Each object is responsible for a specific task. An object is an encapsulation of the Data values (Objects S) and Operation (Methods, Object Function). Some important features in object-oriented programming are described below. Two of these features, reusability and persistency will be described in the next section.

Messages and methods

An action is initiated in object-oriented programming by the transmission of a message to an object responsible for the action. The object to whom the message is sent perform some method to satisfy the request.

Classes and instance

A class is a template used to create new objects. A class encompasses both data and methods. All objects are instances of a class. All objects belonging to the same class use the same methods in response to similar messages. The definition of the class determines for which messages the object has a method.

Inheritance

Classes can be organized into a hierarchical inheritance structure. A subclass inherits attributes and methods from a superclass. This is the ability to pass along class properties from one class to another. The resulting subclass has all the properties of its parent, which is the superclass.

4 Reusability and persistence

One of the goal of the object-oriented programming is to reuse objects. Reusability is the ability of software products to be reused, in whole or in part, for a new application (Biggerstaff and Perlis, 1989).

One of the dreams of the statistical designer community is to have widely used libraries of reusable software. Object-oriented programming systems, languages, and architectures have brought this dream closer. Common-sense organizational principles like reusability and interchangeability are still exception rather than the rule. Indeed, it is difficult to extract element of software from one project that can be easily used in an unrelated project, because each portion of the code has interdependency with all other portions of the code. This interdependency may result of data defining or may be functional dependency. It is important to be able to manage the *objects* by taking into account this functional dependency. One of the dreams of the statistical designer community is to have a widely used libraries of reusable software. Object-oriented programming systems, languages, and architectures have brought this dream closer. Common-sense organizational principles like reusability and interchangeability are still exception rather than the rule.

Persistency is the ability of an object to continue to exist outside of the execution time of programs that manipulate the objects.

During a S session, the designer creates and manipulates object. It may be necessary to keep these objects between sessions. The idea of the S approach is that objects assigned at the top level become permanent and can be accessed during later sessions. S allows multiple directories in a search path. This is useful for organizing functions and data objects by projects. The S approach takes less organization on the part of the user and also prevents important values from being lost by mistake. However, it can waste space by saving unnecessary objects. Ideally, programs would just create objects as they please and little by little, objects that are not used would vanish into far-away abysses. But it isn't so. Therefore, we need a

storage and retrieval facility for the object.

Since the call of the different object functions is hierarchical, the most suitable and natural way to store the objects in S is to use a hierarchical structure. The hierarchy of a S function is the list of all the other functions called directly or indirectly by the S function.

Object-oriented systems also need an efficient garbage collector. A functionality of such garbage collector is the dynamic allocation of data objects (Hayes, 1991). Those few objects which survive for a while, however, are likely to survive considerably longer.

5 Commands

We now describe some commands which partially solve the problems addressed above. The commands may be divided into three groups depending on whether they:

1. help the user to *organize* his S functions,
2. allow users to *exchange* S functions, or
3. perform some *cleaning* tasks.

The commands are mainly based on a program called `mklist` which gives the hierarchy of a S function. By that, we mean the list of all the other functions called directly or indirectly by the S function. The kind of list produced by `mklist` depends on whether it is used by the user to understand the function structure or by another command to perform a specific task. `mklist` may be called from the Unix shell or in a S session but the syntax is different as the case may be. When `mklist` is called from the shell, the synopsis is:

```
mklist -[a|d|h|w] -o S function name
```

The option a, d, h or s refers to the kind of list the user wants: alphabetical (a), without redundancy (s) or showing the structure of the hierarchy in depth (d) or horizontally (h). The list is written on standard output. In a S session, the calling sequence is:

```
mklist(fun, type="d")
```

If type is "w" or "a", the value returned by `mklist` is a character vector containing the names of the functions in the hierarchy of fun.

In the other cases, the hierarchy's structure is

displayed on the screen.

We now describe the commands. There are eight commands that a user can call within a S session. In the description below, `fun` and `dir` are character strings giving the name of a function or a directory while `list` is a character vector containing the names of S functions or directories.

1. Commands to organize

* `mkdirfun(fun, dir)`

`mkdirfun` creates a function subdirectory `dir` within the current S .Data directory to contain the S function `fun` and its hierarchy. The directory is added to the S search list.

* `rmdirfun(list)`

For each directory in `list`, `rmdirfun` checks for its existence and if so, the directory is deleted along with its content. The directory is also removed from the S search list.

* `mvfun(fun, dir)`

`mvfun` moves the S function `fun` and its hierarchy to the function subdirectory `dir`. When `dir` does not exist, it is created.

* `lsdir(list)`

`lsdir` perform a task similar to the standard S function `ls`. For each function directory in `list`, the names of the functions in the directory are put in a character vector. The value returned by `lsdir` is the list of all the character vectors. When `lsdir` is called without arguments, it produces a character vector of all the function subdirectory names within the current S .Data. This is useful if one wants to add these directories to the S search list.

2. Exchange commands

* `export(fun, archive, type="t")`

`export` creates an archive file for S function `fun`. The archive contains the source code of `fun` and its hierarchy. Each function of the hierarchy has its source in a separate file/of the archive. The latter may be of type `tar` (`type="t"`) or `cpio` (`type="c"`). The archive file is created in the current Unix directory from which S is

called. A user can retrieve its contents using the `import` command below.

* `import (archive, dir)`

`import` opens an archive file previously created by `export` and puts its content in the function subdirectory `dir`.

3. Commands to clean

* `drop(list)`

Each function in `list` is deleted along with its hierarchy.

* `keep(list)`

Every function of the current `S .Data` is deleted except those listed in `list` which are kept with their hierarchy.

6 Conclusions

We have described a set of tools for managing `S` objects. The goal achieved by those tools is the portability between systems and the creation and archiving of libraries of objects.

The `S` objects are stored in a library which is organized in a hierarchy. The hierarchy of an `S` object is composed of the list of all the objects called directly or indirectly by this object. This type of structure helps in managing the objects and facilitate object reuse. The other advantage of this model for managing object is that there are automatic procedures to create and archive objects. The management of objects is done by the various tools that accesses the library.

Memory management is an important issue when dealing with object storage and retrieval. Hence an efficient garbage collector with dynamically-allocated data is essential and should be part of any archiving system. The garbage collector should also use the information concerning the archiving date to reorganize memory.

This resulting archiving system is portable from the designer to the user. The user can reuse part or all the objects created by the designer in the context of his specific application. Also, objects that were developed for a specific project can be easily accessed and reused in other projects. Having an efficient archiving system makes the task of the user easier and also cuts down on the development time of new systems. Finally, the main drawback

of software systems is that they are used for a specific application and in a particular context but they cannot easily be reused in another context. Considering the time it takes to build an entire system from scratch, the approach presented provides a good solution to the problem of code that are used only once and then discarded when the application becomes obsolete.

The libraries of objects and the associated tools for managing the objects is an infrastructure which can be used by organizations to help them structure software and to reuse software within a project or between projects. In large organizations, there may be several libraries of reusable software.

We plan to enhance the various tools that were developed for managing `S` objects and provide an efficient real-time garbage collector for the archiving system based on the lifetime of objects. We will also extend the tools for managing New `S` objects (Chambers and Hastie, 1991).

7 References

- Becker, Richard A., Chambers John M. and Allan R. Wilks [1988], *The New S Language, A Programming Environment for Data Analysis and Graphics*, Wadsworth & Brooks/Cole Computers Science Series, John Wiley & Sons, New York.
- Biggerstaff, Ted and Alan Perlis (eds) [1989], *Software Reusability*, Volume 1 & 2, ACM Press, New-York.
- Budd, Timothy [1991], *An Introduction to Object-Oriented Programming*, Addison-Wesley, New York.
- Chambers, John M. and Trevor J. Hastie (eds.) [1991], *Statistical Models in S*, Wadsworth & Brooks/Cole Computers Science Series, John Wiley & Sons, New York.
- Hayes, Barry [1991], *Using Key Object-Oriented Opportunism to Collect Old Objects*, in *Object-Oriented Programming: Systems Languages and Applications*, p. 33-46
- Tierney, Luc [1990], *LISP-STAT: An Object-Oriented Environment for Statistical Computing and Dynamic Graphics*, John Wiley & Sons, New York.

A Selection of Utilities to Facilitate Data Analysis Under UNIX

Lionel Galway and Daniel Relles,
RAND, Santa Monica, CA.

Abstract

The steadily increasing power of desktop UNIX¹ workstations and the vast improvements in local mass storage capacity have made it feasible and inexpensive for researchers to locally store and analyze large data sets. In the course of several policy analysis projects at RAND which required substantial data analysis of a variety of data sets, we developed a set of utilities which work well with standard UNIX utilities to facilitate such analyses.

The first set of tools supports operations on column-oriented data. Such data sets are very common, particularly as extracts from other data systems. UNIX tools such as *awk*, *grep*, *sed*, etc. do not handle such data gracefully, since they are oriented toward data with specific field separators. We describe three simple filters which, together with a simple text dictionary describing the data, provide a researcher with convenient, easily-used facilities for sorting, subsetting, and extracting specified fields from column-oriented data sets.

We also present two tools which are useful for exploratory data analysis. One provides a succinct picture of the values taken by each variable in a single data set, while the other provides a useful way of comparing two related data sets (e.g. successive years of data).

Introduction

RAND conducts policy research in a wide variety of areas: military logistics, health care economics, demographics, etc. Much of this work makes heavy use of data analysis. Like most of the rest of the statistical community, we have moved most of the analysis that was previously done on mainframes to desktop workstations running UNIX. In the process we have developed a number of utilities to make our work easier in the new environment and to take advantage of low-cost, abundant computer power.

In this paper we present two sets of utilities. The utilities in the first set are designed to manipulate column-oriented data in the UNIX environment (column-oriented data defines variables by column position in each record). The second set of utilities is useful for exploring new data sets. Our aim has been to make tools which are simple and easy to use, but which can also be combined to be very powerful. In this paper we will emphasize the functional aspects of the utilities, not implementation details.

Column-oriented data in UNIX

Most users of UNIX are familiar with the powerful set of text utilities which have been developed in UNIX. These

include classics such as *grep*, *awk*, and *sed*, and newer programs such as *perl*. These programs provide powerful capabilities for manipulating textual data. However, most of these utilities have the drawback that they are designed for free-format data, i.e. the fields within a record are delimited by a specified character, whitespace (tab or space) by default. Most of them can handle column-oriented data, but at the expense of somewhat clumsy syntax and only by column number. This puts the burden on the analyst of being forced to translate variable names into column numbers whenever any manipulations need to be done on the data. In addition, different UNIX utilities use different conventions for counting columns and specifying column ranges.

Unfortunately, many data sets (particularly those which are extracts of data maintained on mainframes) are column-oriented. We built a set of utilities which are called from the command line, and allow the user to perform operations on fields denoted by names, rather than numbers.

The cornerstone of these programs is a dictionary (an ASCII file) that maps field names into column positions. Such a dictionary can be prepared with a simple text editor when a data set is acquired and then provides an error-free way of translating a field name into the corresponding columns. This dictionary provides the foundation for all of the utilities discussed in this paper.

An example of such a dictionary is shown below for a set of repair data from a U.S. Navy depot.

date	i	1	5	date of action
nsn	c	6	17	ntl stock num
fsc	i	6	9	fed stock id
niin	i	10	17	ntl inven num
acttyp	i	21	22	action type
qtytrans	i	40	46	quantity

The first entry in each line of the dictionary contains the name of the field, followed by the field's starting and ending columns, counting from one. (The "i" or "c" indicates whether the field is numeric or character, and is used by some other programs which also access this dictionary). Any characters beyond the ending column on each line are ignored and can be used for comments. Note that the definition of fields may overlap: the national stock number (nsn) may be accessed as a whole or as the federal stock class (fsc) and national item inventory number (niin).

We have found that three simple utilities provide great flexibility and power. *Stawk* extracts the contents of a group of selected fields for each record and writes them to standard output separated by a single space. The output of *stawk* is therefore in an appropriate form to be used by the usual field-oriented UNIX utilities.

¹UNIX is a trademark of AT&T/Bell Laboratories.

The syntax of the command is illustrated below:

```
stawk navy.dct navy.dat fsc date acttyp
```

The first two arguments are the dictionary and data file, while the remaining arguments are the selected field names. This syntax is common to all of the utilities.

In addition to explicit field names, *stawk* interprets an argument of "#" as indicating the selection of the entire original record. To allow more flexibility in the output of *stawk*, the whitespace between fields can be suppressed by appending a "-" to individual field names.

Stgrep selects or excludes records based on the value of a single field:

```
stgrep {-v} navy.dct navy.dat fsc  
goodfsc
```

The argument after the data file indicates the field whose values are to be specified; only one field can be so designated. The fourth argument is taken to be a file of possible values for *fsc*, or, if no such file exists, is interpreted as a single value for *fsc*. The optional "-v" switch is used to indicate that *stgrep* should exclude records. All records meeting the criterion are written out (or excluded) in their entirety to standard output (logical ands can thus be constructed by piping several *stgreps* together).

Stsort sorts a data set by field:

```
stsort navy.dct navy.dat niin+ date+  
acttyp-
```

Stsort is an interface to the UNIX *sort* command and simply translates the field names into appropriate column arguments for *sort*. "+" (optional) means that fields should be sorted in ascending order, "-" denotes descending order. Note that if fields overlap in the dictionary the *stsort* command may be nonsensical.

The primary use of these commands is to assist in the preparation of data sets for further work. In addition, they can also allow some simple computations to be performed quickly. For example, the following command uses *stawk* to count the number of times each part appears in the Navy database, after *stgrep* has selected a subset of records by federal stock class (*fsc*):

```
stgrep navy.dct navy.at fsc goodfsc |  
stawk navy.dct - niin | sort | uniq -c
```

The result:

```
...  
49      00719403  
4       00719404  
18      00751999  
48      00752227  
...
```

Note that *stawk* is informed that the data set is standard input by using "-" as the data file argument (a standard UNIX convention).

These three utilities are written in C and use only the standard C libraries. *Stawk* and *stgrep* can therefore be ported to almost any computer with a C compiler. *Stsort* is similarly portable, but calls the UNIX *sort* command to do the actual sorting. If it were to be used on another platform an appropriate sort command would have to be constructed from the arguments.

Exploring Data Files: One-digit Frequencies

After acquiring a data set, an analyst usually wants to check out the variables of interest, for example to make sure that the codes are the ones expected, and that there are no coding problems or extraneous characters. Typically, a list of all values for each field is not useful; for continuous variables such as income the list could be very lengthy. Instead, a rough idea of the distribution of values for each field is needed.

One-digit frequencies offer a useful tradeoff between detail and conciseness. The idea is simple: first, for each field within each record, replace the actual value by a code which retains only the most significant digit (e.g. 01025 becomes 01xxx, 00203 becomes 002xx). Then, for each field, count the number of times each coded value appears in the data. For numeric fields, this gives a rough histogram of the values with breaks that are evenly spaced on a logarithmic scale.

The implementation of one-digit frequencies is also simple:

```
fdigit navy.dct navy.dat |sort|uniq -c
```

For each field, in each record, *fdigit* writes out a single line consisting of the field name, and its coded value:

```
...  
transtyp          1x  
qtytrans          01xxx  
...  
transtyp          2x  
qtytrans          02xxx  
...
```

The sorting and counting are done by the UNIX *sort* and *uniq* commands to produce (with a little followup formatting) a one-digit frequency table. Displayed next is a part of the one-digit frequency table for the variable *qtytrans* from the Navy data:

```

qtytrans          40.
qtytrans          00000    127.
qtytrans          00001    47115.
qtytrans          00002    4213.
...
qtytrans          0001x    3752.

```

Qtytrans indicates how many parts were requested by repair workers in a single order. Somewhat surprisingly, most of the orders are for only a single unit. We also note that some of the orders have blank or zero quantities; these records should be examined more closely.

One-digit frequencies are most useful for numeric variables such as qtytrans which have many different values. However, for alphanumeric codes with relatively few values they can also be informative.

Exploring Data Files: Statistical File Differencing

Research projects often find themselves in the situation of receiving several sets of similar data. In multi-year research projects, for example, data may become available quarterly or annually. In other cases a corrected data set may be provided some time after the original one is obtained. In all of these situations the question arises of how the new data set differs from the old one. Dates should certainly be different in an update data set but coding conventions might also change or new features of the data may reflect exogenous events such as the introduction of new regulations. When a data set consists of tens or hundreds of variables, it becomes important to rank the variables by their amount of change and then focus attention on the ones which have changed the most.

One approach to this problem is to compare one-digit frequencies. Below, segments of the one-digit frequency outputs for the variable "outlier" from two successive years of Medicare payment data are combined in a single table:

var	code	1988	1989
outlier		30162	1342105
outlier	0	1677259	593796
outlier	1	57387	38061
outlier	2	21444	10975
...			

Outlier indicates whether or not Medicare paid more than the "standard" cost for a patient's condition. It is a one-digit code and so the one-digit frequency codes are just the value of the code itself. The counts certainly appear different from 1988 to 1989, but we should take into account such things as the relative number of patients in each file. Also, we have no way of knowing whether outlier has changed more or less than other variables in the data.

The key to measuring the change for outlier between 1989 and 1990 is the recognition that the two right-hand

columns have the form of a $2 \times K$ contingency table, where K is the number of different levels of outlier in both files. Now we can formulate the question "Is there a change?" in a slightly different way: "Are the proportions equal?" This new question is easily addressed by adapting standard methods from contingency table analysis.

We construct our measure of change in three steps:

(1) For each of M variables in the data set, construct a $2 \times K_m$ contingency table of counts (K_m varies from field to field) and calculate Pearson's chi-square statistic for the model of equal proportions (independence).¹

(2) Normalize each chi-square statistic by its "degrees of freedom", $K_m - 1$. This is done to avoid "overvaluing" variables with many different codes.

(3) Rank the variables in descending order by the chi-square statistic. The variables which deviate most from equal proportions will be at the top of the list.

In the Medicare data outlier ranks number two in the list of variables ordered by chi-square statistic, with a chi-square of 257616 (number one was a year variable, with a value of 475178, while number three had a value of only 35380). To better illustrate just how much the distribution across codes had changed for outlier from 1988 to 1989, the table below reproduces the one-digit frequencies from the table above, with the addition of the fraction of records in each year which fall into each code category, expressed as a percentage of the total records within each year:

var		1988	%	1989	%
outlier		30162	2	1342105	65
outlier	0	1677259	86	593796	29
outlier	1	57387	3	38061	2
outlier	2	21444	1	10975	1
...					

Further investigation of the data and its sources revealed that there was a change in the usage of the first two values of outlier (blank and 0) and also that new Medicare policies reduced the number of patients for whom various types of outlier payments were made.

Summary

Researchers at RAND have made use of all of these utilities. *Stgrep*, *stawk* and *stsort* have been used for data preparation by a number of projects and with a wide range of data set sizes. One-digit frequencies and statistical file differencing take advantage of very cheap workstation-based computing to compute exploratory statistics that help researchers get insight into their data at the beginning of analyses.

All of these programs are simple in concept. *Stgrep*, *stawk* and *stsort* are written in C; one-digit frequencies and

¹See, e.g. Fienberg, S.E., *The Analysis of Cross-classified Categorical Data*, MIT Press, 1981, p. 40.

statistical file differencing are implemented by FORTRAN-77 programs and UNIX shell scripts. Source code and scripts are available from the authors via email to Lionel_Galway@rand.org.

Design of an S Function for Robust Regression Using Iteratively Reweighted Least Squares

Richard M. Heiberger
Temple University
Philadelphia, PA 19122-2585
rmh@astro.ocis.temple.edu

Richard A. Becker
AT&T Bell Laboratories
Murray Hill, NJ 07974
rab@research.att.com

Abstract

We develop a set of S functions for robust regression using the technique of *Iteratively Reweighted Least Squares (IRLS)*. Together with a set of weight functions, function `rreg` is simple to understand and provides great flexibility for IRLS methods. The discussion focuses on the programming strategies adopted to achieve the twin goals of power and simplicity.

1. Introduction

Least squares regression is sensitive to outlier points. A single point that appears to be different from the others in a data set may have an inordinate influence on the estimated regression coefficients. Techniques that attempt to minimize the effect of outlier points are called *robust* (Huber, 1972, 1981). Section 2 describes one important class of techniques, *Iteratively Reweighted Least Squares (IRLS)*. Section 3 presents a very general S function, `rreg`, for IRLS that allows the user complete flexibility in the choice of weight function. The `rreg` function permits an unlimited number of functional forms with an unlimited number of tuning constants. Section 4 discusses some of the programming strategies adopted to achieve the twin goals of power and simplicity.

2. Iteratively Reweighted Least Squares

The method of Iteratively Reweighted Least Squares is composed of an underlying weighted least squares fit placed inside an iteration loop. At each iteration we carry out a least-squares fit using a set of weights, one weight per observation, constructed by a specified weighting function from the current residuals. (Initial weights are based on residuals from an initial fit, generally unweighted least squares.) The iterative process terminates when the residuals are unchanged on two successive passes. Several alternate convergence criteria may be used instead.

IRLS depends heavily on the choice of a weight func-

tion. The weight functions most commonly used in the literature are listed in Table 1. The definitive weight function has yet to be determined. Computational techniques and weight functions for robust regression are discussed in Coleman, et al (1980), and in UREDA Chapter 11 (Hoaglin, Mosteller, and Tukey 1983).

To begin, we show a simplified version of `rreg`:

```
rreg.1 ← function( x, y, w=rep(1, nrow(x)),
  init=lsfit(x, y, w, int=FALSE)$coef,
  method=wt.bisquare, iter=20,
  acc=10*.Machine$single.eps^.5
)
{
  irls.delta ← function(old, new)
    sqrt( sum((old - new)^2) / sum(old^2) )

  x ← cbind(1,x)

  coef ← init
  resid ← y - x %*% coef

  for(iter in 1:iter) {
    previous ← resid
    scale ← median(abs(resid))/0.6745
    w ← method(resid/scale)
    temp ← lsfit(x, y, w, int=FALSE)
    resid ← temp$residuals
    coef ← temp$coef
    convi ← irls.delta(previous, resid)
    if(convi <= acc) break
  }
  if(convi > acc)
    warning(paste("failed to converge in",
      iter, "steps"))
  list(coef=coef, residuals=resid, w=w)
}
```

This executable S code is very similar to the informal description at the start of the section. It defines a local function, `irls.delta`, to determine convergence, computes initial coefficients and residuals, and then repeatedly computes new weights, uses them to perform a

weighted regression and tests for convergence.

We will initially use `rreg.1` with the bisquare weight function defined by

$$w(u) = \begin{cases} \left(1 - \left(\frac{u}{c}\right)^2\right)^2 & 0 < \left|\frac{u}{c}\right| \leq 1 \\ 0 & 1 < \left|\frac{u}{c}\right| \end{cases}$$

and the S function

```
wt.bisquare ← function( u, c=4.685 )
{
  U ← abs(u/c)
  w ← ((1 + U) * (1 - U))^2
  w[U > 1] ← 0
  w
}
```

3. rreg and its Weight Functions

We now generalize `rreg.1` into the function `rreg` displayed in the appendix. A more extensive discussion appears in Heiberger and Becker (1992). The complete function is distributed with the August 1991 and later releases of S.

3.1 Convergence Control

The first step of generalization is to introduce more complicated weighting functions. The default weight function, `wt.default`, is defined as Huber weights until convergence, then two steps of bisquare weights. The major complication in the construction of the default weight function is that we must allow the weight function to have control over final convergence decisions.

```
wt.default ← function(u)
{ # huber till convergence,
  # then two steps of bisquare
  fr ← sys.parent(1)
  if(!exists("method.doing.bisquare",
             frame = fr))
  {
    assign("method.doing.bisquare", FALSE,
           frame = fr)
    assign("method.in.control", TRUE,
           frame = fr)
  }
  converged ← get("converged", frame = fr)
  # from rreg
  method.doing.bisquare ←
    get("method.doing.bisquare",
```

```
      frame = fr)
  if(converged || method.doing.bisquare)
  {
    w ← wt.bisquare(u)
    if(method.doing.bisquare)
      assign("method.exit", TRUE,
             frame = fr)
    else
      assign("method.doing.bisquare",
             TRUE, frame = fr)
  }
  else w ← wt.huber(u)
  w
}
```

Allowing the weight function to influence convergence decisions requires mild cooperation from the `rreg` function. We arrange for this cooperation by having `rreg` use two objects to control convergence checking. Each of these objects exists in the frame (set of local objects) owned by `rreg`. The weight function can create objects in `rreg`'s frame by using the `assign` function rather than the normal assignment arrow operator.

A weighting method can force iterations to continue after `rreg` has determined convergence by placing a logical object named `method.in.control` in `rreg`'s frame. As long as `method.in.control` is TRUE, iterations will continue (up to the maximum number specified in `rreg`'s `iter` argument).

A method can also decide by itself that convergence has occurred—`rreg` terminates execution if the method sets object `method.exit` to TRUE in `rreg`'s frame.

The default weight function, `wt.default`, uses both of these objects: `method.in.control` is set TRUE so that iterations continue beyond the initial convergence with Huber weights; `method.exit` is set TRUE to terminate iterations after two steps of bisquare weighting. In addition, `wt.default` uses the object `converged` in `rreg`'s frame to determine when to switch from Huber to bisquare weighting.

A weight function may need persistent storage in which to keep values from one call of the function to the next. For example, `wt.default` stores an indicator, `method.doing.bisquare`, of whether it is currently doing the Huber or bisquare weight functions. It arranges for persistent storage by using the same technology as was used for taking control of convergence, by creating objects in `rreg`'s frame. Any values stored in the `rreg` frame will persist over several calls to the weight function, and will vanish when the `rreg` function completes execution. We have adopted the convention that all objects to be assigned into the parent's frame have names

of the form "method.*".

3.2 Alternate Convergence Criteria

Another generalization allows `rreg` to have several alternate convergence criteria. In `rreg.1`, we defined convergence as a test on the relative change of the residual at each iteration step. This default in `rreg` is specified by the optional argument `test.vec="resid"`. Other reasonable criteria are relative change in the coefficients `test.vec="coef"` or in the weights `test.vec="w"`, or an orthogonality test of the residuals and `x` (specified by `test.vec=NULL`). The function used to evaluate convergence depends on which convergence criterion is chosen. We use the `get` and `assign` functions to store the appropriate vectors in a common name, previous, so that the same S expression, using function `irls.delta` can be used for all three vector convergence techniques. A different function, `irls.rxxwr`, is used for the orthogonality test. Both convergence functions are defined local to the `rreg` frame.

3.3 Detecting and Reporting Convergence Status

The `rreg` function reports its convergence status by including two elements in its value list: a vector `conv` containing the values of the convergence criterion at each iteration and a character string `status` telling how the iterations ended. The `status` indicator can be "converged", "ran out of iterations", or "could not compute scale of residuals". The latter situation comes up when the median absolute residual is zero; iterations cannot continue because the scaled residuals cannot be computed.

3.4 Standard Weight Functions

There are several commonly used weighting methods listed in Table 1. They are used with their default tuning constants by calls of the form

```
> rreg(x, y, method=wt.bisquare)
```

They are used with other tuning constants by either

```
> rreg(x, y,
+ method=function(r) wt.bisquare(r,c=3))
```

or

```
> bi.3 ← function(r) wt.bisquare(r,c=3)
> rreg(x, y, method=bi.3)
```

3.5 Multiple Tuning Constants

Since the weight functions are written in the S language, it is easy to construct new weight functions with more than one tuning constant. Thus, we now have a function for the Hampel weights

```
wt.hampel ← function(u, a=2, b=4, c=8)
{
  U ← abs(u)
  A ← (U <= a)           # increasing
  B ← (U > a) & (U <= b)  # flat
  C ← (U > b) & (U <= c)  # descending
  D ← (U > c)             # zero
  w ← u
  w[A] ← 1
  w[B] ← a/U[B]
  w[C] ← a*((c-U[C])/(c-b))/U[C]
  w[D] ← 0
  w
}
```

with three parameters. As with the single tuning constant functions, it is easy to modify the values of multiple tuning constants:

```
> rreg(x, y, method=function(r)
+ wt.hampel(r, a=.1, b=.3, c=.8))
```

4. Programming Strategies

When we designed `rreg`, we based many of our programming decisions upon general goals of simplicity and power. In particular:

We wanted `rreg` to be simple to understand. Because S provides very high level operations, the entire function is reasonably short and easy to read. It is easy for users to modify `rreg` to accommodate special needs.

While retaining simplicity, we still wanted `rreg` to be powerful enough to allow arbitrarily complicated weight functions such as `wt.default`. This is especially important since there is no one weight function for robust regression that is universally accepted. Because S functions are first class objects, they were a natural way to express the weight computations. The ability of the weight functions to encapsulate their own parameters (tuning constants, etc.) further simplified the basic `rreg` arguments.

A subsidiary goal for the weight functions was to make it simple to express a simple weight function. The examples of `wt.bisquare` and `wt.hampel` illustrate this.

4.1 Persistent Storage

The mild cooperation of providing a restricted part of `rreg`'s name space (all names of the form "method.*") to the weighting method gives the method the maximum amount of control with the least work on the part of `rreg`. The `rreg` function needs to know who is in charge of convergence, nothing more. It guarantees not use any object names in the restricted name space.

4.2 Scaling of Residuals

We have defined the weight functions to work on scaled variables with location parameter 0 and scale parameter 1. For many weighting functions changes in this assumption can be made by adjusting the tuning constants. Otherwise, new weight functions can be defined with different assumptions.

4.3 Local Functions

We have defined the convergence functions in the frame of `rreg`. They do not seem general enough to make part of the public name space of objects, and yet they do carry out a specific computation that is appropriate to encapsulate in a function. By isolating the convergence computations in functions, we simplify the main loop of `rreg`, making it easier to understand and modify.

4.4 Protection from Division by Zero

The `irls.delta` function introduced in `rreg.1` is simple and straightforward. A substantially more complicated version is required in `rreg` in order to provide protection against exponential overflow and division by zero.

A third version, tailored for machines that support IEEE arithmetic, provides the same protection more naturally:

```
irls.delta.ieee ← function(old, new) {
  result ←
    sqrt(sum((old - new)^2)/sum(old^2))
  if(is.na(result)) 0 else result
}
```

4.5 Lazy Evaluation

From a first reading of the `rreg` code, it seems that the argument `w` of `rreg` is first referenced in the body of the function when it is assigned a value in the loop. The subtlety of lazy evaluation is that the initial value of `w` is referenced in the call to `lsfit` that occurs when the

default value for argument `init` is evaluated.

References

- Becker, R. A., J. M. Chambers, and A. R. Wilks (1988), *The New S Language: A Programming Environment for Data Analysis and Graphics*, Wadsworth, Monterey, CA.
- Coleman, D., Holland, P., Kaden, N., Klema, V., and Peters, S. C. (1980), "A system of subroutines for iteratively re-weighted least-squares computations", *ACM Trans. Math. Soft.*, Vol. 6, pp. 327-336.
- Heiberger, Richard M., and Richard A. Becker (1992, to appear). "An S Function for Iteratively Reweighted Least Squares," *Journal of Computational and Graphical Statistics*.
- Hoaglin, D. C., Mosteller, F. and Tukey, J. W. (1983), *Understanding Robust and Exploratory Data Analysis*, Wiley, New York.
- Huber, P. J. (1972), "Robust Statistics: a Review", *Annals of Mathematical Statistics*, 43, 1041-1067.
- Huber, P. J. (1981), *Robust Statistics*, Wiley, New York.

Table 1: Weight functions provided as part of the S implementation of `rreg`.

Estimator	S function
andrews	wt.andrews(u, c = 1.339)
bisquare	wt.bisquare(u, c = 4.685)
cauchy	wt.cauchy(u, c = 2.385)
default	wt.default(u)
fair	wt.fair(u, c = 1.4)
hampel	wt.hampel(u, a = 2, b = 4, c = 8)
huber	wt.huber(u, c = 1.345)
logistic	wt.logistic(u, c = 1.205)
median	wt.median(u)
talworth	wt.talworth(u, c = 2.795)
welsch	wt.welsch(u, c = 2.985)

Appendix: The rreg Function

```

rreg ← function(x, y, w = rep(1, nrow(x)),
  int = TRUE,
  init = lsfit(x, y, w, int = FALSE)$coef,
  method = wt.default,
  wx, iter = 20,
  acc = 10 * .Machine$single.eps-.5,
  test.vec = "resid")
{
  irls.delta ← function(old, new)
  {
    a ← sum((old - new)2)
    b ← sum(old2)
    if(b >= 1 ||
      a < b * .Machine$double.xmax
      sqrt(a/b)
    else .Machine$double.xmax
  }
  irls.rwxr ← function(x, w, r) {
    w ← sqrt(w)
    max(abs((as.vector(r * w) %>% x) /
      sqrt(as.vector(w) %>% (x2))))
    /sqrt(sum(w * r2))
  }
  if(!(any(test.vec ==
    c("resid", "coef", "w", "NULL")) ||
    is.null(test.vec)))
    stop("invalid test.vec")
  if(int) x ← cbind(1, x)
  else x ← as.matrix(x)
  if(!missing(wx)) {
    if(length(wx) != nrow(x))
      stop("Length of wx must equal
        number of observations")
    if(any(wx < 0))
      stop("Negative wx value")
    w ← w * wx
  }
  coef ← init
  if(ncol(x) != length(coef))
    stop("Must have same number of
      initial values as coefficients")

  resid ← y - x %>% coef
  converged ← FALSE
  status ← "converged"
  conv ← NULL
  method.in.control ← method.exit ← FALSE

```

```

for(iter in 1:iter) {
  if(!is.null(test.vec))
    previous ← get(test.vec)
  scale ← median(abs(resid))/0.6745
  if(scale == 0) {
    convi ← 0
    method.exit ← TRUE
    status ← "could not
      compute scale of
      residuals"
  }
  else {
    w ← method(resid/scale)
    if(!missing(wx))
      w ← w * wx
    temp ← lsfit(x, y, w,
      int = FALSE)
    coef ← temp$coef
    resid ← temp$residuals
    if(!is.null(test.vec))
      convi ← irls.delta(
        previous,
        get(test.vec))
    else convi ←
      irls.rwxr(x, w, resid)
  }
  conv ← c(conv, convi)
  converged ← convi <= acc
  done ← method.exit ||
    (converged && !method.in.control)
  if(done) break
}
if(!done)
  warning(status ← paste(
    "failed to converge in",
    iter, "steps"))
if(!missing(wx)) {
  tmp ← (wx != 0)
  w[tmp] ← w[tmp]/wx[tmp]
}
list(coef = coef, residuals = resid,
  w = w, int = int, conv = conv,
  status = status)
}

```

STATISTICAL COMPUTATION USING GAUSS: EXAMPLES IN PROCESS CAPABILITY RESEARCH

Ken Hung and Daniel Hagen
College of Business and Economics
Western Washington University
Bellingham, WA 98225

Abstract

Examples are given which demonstrate that numerical computations in statistical research can be executed efficiently using GAUSS.

1. Introduction

The process capability index, C_p , is defined as follows:

$$C_p = (USL - LSL)/6\sigma \quad (1)$$

where USL = upper specification limit, LSL = lower specification limit and 6σ = natural tolerance. See Kane(1986). The natural estimator, \hat{C}_p , is:

$$\hat{C}_p = (USL - LSL)/6s \quad (2)$$

where s , a sample standard deviation, is an estimator for σ . The mean of \hat{C}_p is given by Chan et al. (1988):

$$E(\hat{C}_p) = C_{11} C_p \quad (3)$$

$$\text{where } C_{11} = \sqrt{\frac{n-1}{2}} \frac{\Gamma((n-2)/2)}{\Gamma((n-1)/2)}$$

as defined in Chou and Owen (1989). It can be shown by direct integration that

$$E(\hat{C}_p^2) = \frac{(n-1)}{(n-3)} C_p^2 \quad (4)$$

Thus, the variance of \hat{C}_p is given by

$$V(\hat{C}_p) = \left(\frac{(n-1)}{(n-3)} - C_{11}^2 \right) C_p^2 \quad (5)$$

See Chou and Owen (1989). These are derived under the assumption of normality.

The effect of nonnormality on the expectation and variance of \hat{C}_p can be studied via

$$E(\hat{C}_p) = \sqrt{\frac{n-1}{2}} \sum_{i=0}^n A B \quad (6)$$

where

$$A = \frac{n!}{i!(n-i)!} p^i (1-p)^{n-i}$$

and

$$B = e^{-\lambda/2} \sum_{l=0}^{\infty} \left(\frac{\lambda}{2}\right)^l \frac{1}{l!} \frac{\Gamma((n-2)/2+l)}{\Gamma((n-1)/2+l)}$$

where $\lambda = (i(n-i)/n)a$.

In the case when $p=1$, (6) reduces to C_{11} defined in Chou and Owen (1989). Also

$$E(\hat{C}_p^2) = \frac{n-1}{2} \sum_{i=0}^n A B' \quad (7)$$

where

$$B' = e^{-\lambda/2} \sum_{l=0}^{\infty} \left(\frac{\lambda}{2}\right)^l \frac{1}{l!} \frac{\Gamma((n-3)/2+l)}{\Gamma((n-1)/2+l)}$$

Two special cases of interest arise that (7) reduces to $(n-1)/(n-3)$ when $p = 1$ and (7) reduces to C_p^2 when $n \rightarrow \infty$.

A generalization of (6) and (7) can be given below by (8) and (9). See Kotz and Johnson (1991).

$$E(\hat{C}_p) = \sqrt{\frac{n-1}{2}} \sum_{i=0}^n \sum_{j=0}^{n-i} C D \quad (8)$$

where

$$C = \frac{n!}{i!j!(n-i-j)!} p_1^i p_2^j (1-p_1-p_2)^{n-i-j}$$

and

$$D = e^{-\lambda/2} \sum_{l=0}^{\infty} \left(\frac{\lambda}{2}\right)^l \frac{1}{l!} \frac{\Gamma((n-2)/2+l)}{\Gamma((n-1)/2+l)}$$

and where $\lambda = ((i+j)-(i-j)^2/n)a$,
also

$$E(\hat{C}_p^2) = \frac{n-1}{2} \sum_{i=0}^n \sum_{j=0}^{n-i} C D' \quad (9)$$

where

$$D' = e^{-\lambda/2} \sum_{l=0}^{\infty} \left(\frac{\lambda}{2}\right)^l \frac{1}{l!} \frac{\Gamma((n-3)/2+l)}{\Gamma((n-1)/2+l)}$$

2. GAUSS Programs

The effect of nonnormality is measured through the parameters ρ and $a = (\mu_1 - \mu_2)^2 / \sigma^2$ where $|\mu_1 - \mu_2| = \sqrt{a}\sigma$. The programs are written with specified values of n , ρ and a for (6) and (8). (7) and (9) differ from (6) and (8)

by a factor of $\sqrt{\frac{n-1}{2}}$. A gamma function of $(n-2)/2$ in

(6) and (8) is replaced by $(n-3)/2$ in (7) and (9).

$V(\hat{C}_\rho)$ can be obtained from $E(\hat{C}_\rho^2) - (E(\hat{C}_\rho))^2$.

A special case is treated in (8) and (9) where $\mu_1 = -\sqrt{a}\sigma$, $\mu_2 = 0$, $\mu_3 = \sqrt{a}\sigma$, $\rho_1 = \rho_2$ and $\rho_3 = 1 - \rho_1 - \rho_2$.

The program listing for (6) and (7) is as follows:

```
g=1;
resmat=0^0^0^0; resmat2=resmat;
```

```
begynne:
```

```
if g le 6; n=10; else; n=30; endif;
```

```
if g==1 or g==7; a=.01; p=.1;
elseif g==2 or g==8; a=.01; p=.5;
elseif g==3 or g==9; a=.25; p=.1;
elseif g==4 or g==10; a=.25; p=.5;
elseif g==5 or g==11; a=1; p=.1;
elseif g==6 or g==12; a=1; p=.5;
elseif g eq 13; goto bottom;
endif;
```

```
ezv=0; ezv2=0; i=0; j=0; L=seqa(0,1,301);
```

```
starti:
```

```
part1=(n!/(i!*(n-i)!))*(p^i*(1-p)^(n-i));
t1=(i*(n-i)/n)*a;
gdenom=gamma(((n-1)/2)+L);
partL=sumc(((t1/2).^L).*(1./L!)).*(gamma(((n-2)/2)+L))./gdenom);
```

```
part2=exp(-(.5*t1))*partL;
ezs((((n-1)/2)^.5)*part1)*part2;
print i|ezs; i=i+1;
if i le n; ezv=ezv | ezs; goto starti;
elseif (i eq (n+1)); ez = sumc(ezv);
endif;
```

```
nrow = n^a^p^ez; resmat = resmat | nrow;
```

```
ezv=0; ezv2=0; i=0; j=0; L=seqa(0,1,301);
```

```
starti2:
```

```
part1=(n!/(i!*(n-i)!))*(p^i*(1-p)^(n-i));
t1=(i*(n-i)/n)*a;
gdenom=gamma(((n-1)/2)+L);
partL=sumc(((t1/2).^L).*(1./L!)).*(gamma(((n-3)/2)+L))./gdenom);
part2=exp(-(.5*t1))*partL;
ezs((((n-1)/2)^.5)*part1)*part2;
print i | ezs; i=i+1;
if i le n; ezv=ezv | ezs; goto starti2;
elseif (i eq (n+1)); ez2 = sumc(ezv);
endif;
```

```
nrow2 = n^a^p^ez2; resmat2 = resmat2 | nrow2;
```

```
print "DONE WITH PARAMETER SET" g;
g=g+1;
goto begynne;
```

```
bottom:
```

```
lprint "Results for E(C_p): n^a^p^ez";
lprint resmat;
lprint "Results for E(C_p^2): n^a^p^ez2";
lprint resmat2;
```

```
lprint "DONE!";
```

The program listing for (8) and (9) is printed below:

```
g=1;
resmat=0^0^0^0; resmat2=resmat;
```

```
begynne:
```

```
if g le 6; n=10; else; n=30; endif;
if g==1 or g==7; a=.005; p1=.05; p2=p1;
elseif g==2 or g==8; a=.005; p1=.25; p2=p1;
elseif g==3 or g==9; a=.125; p1=.05; p2=p1;
elseif g==4 or g==10; a=.125; p1=.25; p2=p1;
```

```

elseif g==5 or g==11; a=.5; p1=.05;p2=p1;
elseif g==6 or g==12; a=.5; p1=.25;p2=p1;
elseif g eq 13; goto bottom;
endif;

ezv=0; ezv2=0; i=0; j=0; L=seqa(0,1,301);
startj:
part1=(n!/(i!*j!(n-i-j)!))*(p1^i*p2^j*(1-p1-p2)^(n-i-j));
t1=((i+j)-(i-j)^2/n)*a;
gdenom=gamma(((n-1)/2)+L);
partL=sumc(((t1/2).^L).*(1./L!).*((gamma(((n-2)/2)+L))./gdenom)));
part2=exp(-(.5*t1))*partL;
ezs=(((n-1)/2)^.5)*part1*part2;
print i | j | ezs; j=j+1;
if j le (n-i); ezv=ezv | ezs; goto startj;
elseif (j gt (n-i)) and (i lt n); i=i+1; j=0; goto startj;
elseif (j gt (n-i)) and (i eq n); ez = sumc(ezv);
nrow = n^a~p1^ez; resmat = resmat | nrow;
endif;

```

```

ezv=0; ezv2=0; i=0; j=0; L=seqa(0,1,301);

startj2:
part1=(n!/(i!*j!(n-i-j)!))*(p1^i*p2^j*(1-p1-p2)^(n-i-j));
t1=((i+j)-(i-j)^2/n)*a;
gdenom=gamma(((n-1)/2)+L);
partL=sumc(((t1/2).^L).*(1./L!).*((gamma(((n-3)/2)+L))./gdenom)));
part2=exp(-(.5*t1))*partL;
ezs=(((n-1)/2))*part1*part2;
print i | j | ezs; j=j+1;
if j le (n-i); ezv2=ezv2 | ezs; goto startj2;
elseif (j gt (n-i)) and (i lt n); i=i+1; j=0; goto startj2;
elseif (j gt (n-i)) and (i eq n); ez2= sumc(ezv2);
nrow2= n^a~p1^ez2; resmat2= resmat2 | nrow2;
endif;

```

```

print "DONE WITH PARAMETER SET" g;
g=g+1;
goto begynne;
bottom:
lprint "Results for E(Cp): n^a~p^ez";
lprint resmat;
lprint "Results for E(Cp^2): n^a~p^ez2";
lprint resmat2;

lprint "DONE!";

```

3. Numerical Outputs

The numerical outputs for (6) and (7) are tabulated below with the reference vlaue under normality.

n	a	p	$E(\hat{C}_p)$	$V(\hat{C}_p)$
10	Normal		1.094	.0884
	.01	.1	1.09375	.08827
	.01	.5	1.091807	.0892076
	.25	.1	1.0822188	.086484
	.25	.5	1.0602702	.0839479
	1	.1	1.0492831	.0821736
	1	.5	0.9750284	.0691651
	Normal		1.027	.0197
30	Normal		1.027	.0197
	.01	.1	1.0263641	.0196851
	.01	.5	1.0255446	.0196534
	.25	.1	1.0154950	.0192908
	.25	.5	0.99608587	.0184851
	1	.1	0.98393207	.0183716
	1	.5	0.91753086	.01516154
	Normal		1.027	.0197

The numerical outputs for (8) and (9) are tabulated below with the reference value under normality.

n	a	p_1	$E(\hat{C}_p)$	$V(\hat{C}_p)$
10	Normal		1.094	.0884
	.005	.05	1.093968	.08830537
	.005	.25	1.091810	.08920817
	.125	.05	1.087501	.08729213
	.125	.25	1.060449	.08406916
	.5	.05	1.068395	.08462937
	.5	.25	0.976603	.07046849
	Normal		1.027	.0197
30	Normal		1.027	.0197
	.005	.05	1.0265693	.01969287
	.005	.25	1.0255446	.01965367
	.125	.05	1.0204805	.01946875
	.125	.25	0.99612686	.01851467
	.5	.05	1.0022662	.01889906
	.5	.25	0.91798395	.01547120
	Normal		1.027	.0197

References:

Bissell, A.F. (1990), How reliable is your capability index, *Applied Statistics*, 39(3), pp. 331-340.

Boyles, R.A. (1991), The Taguchi capability index. *Journal of Quality Technology*, 23(1), pp. 17-26.

Chan, L.K. Cheng, S.W. and Spiring, F.A. (1988), A new measure of process capability C_{pm} , *Journal of Quality Technology*, 20(3), pp. 162-175.

Chou, Y.M., Owen D.B. (1989), On the distribution of the estimated process capability indices, *Communications in Statistics - Theory and Methods*, 18(12), pp. 4549-4560.

Kane, V.E. (1986), Process Capability Indices, *Journal of Quality Technology*, 18(1), pp. 41-52.

Kotz, S. and Johnson, N.L. (1991). Process capability indices for non-normal populations, *Internal Report*, Dept. of Statistics, University of North Carolina, Chapel Hill.

Visualizing Experimental Designs with LISP-STAT

Philip W. Iversen
Department of Statistics
Iowa State University
Ames, Iowa 50011

Mervyn G. Marasinghe
Department of Statistics
Iowa State University
Ames, Iowa 50011

Abstract

The structure diagram described by Taylor and Hilton (American Statistician, 1981) provides a visual display of the relationships between factors for balanced complete experimental designs. Using the idea of factor sets one can obtain the model and the ANOVA table, including expected mean squares, from the structure diagram. This procedure has been implemented in LISP-STAT using a software representation of the experimental design.

1. Introduction

This paper describes a system for visualizing balanced experimental designs. Given information about the factors in a design, the system will produce:

- A Hasse diagram of the nesting relationships among the factors.
- The terms in an appropriate linear model.
- An outline of the ANOVA table.
- Formulas for the expected mean square of each term in the model.

The system uses rules presented in Taylor & Hilton and the object-oriented and dynamic graphics programming features available in LISP-STAT (Tierney, 1990). Section two describes response structures and their relationship with structure (or Hasse) diagrams. Section three summarizes factor sets from Taylor & Hilton, and section four describes the system and the underlying software representation.

2. Response Structures and Structure Diagrams

Responses that are outcomes of a designed experiment can be classified by factors of the experiment. The factors may consist of experimental treatments, or may correspond to various spatial or temporal arrangements of the experimental units, such as plots and blocks. Levels of factors partition the set of responses into disjoint,

nonempty subsets. The set of responses and a corresponding set of factors are said to form a **response structure**. A response structure in which every possible combination of the set of factors is nonempty and all combinations of every subset of factors contain the same number of responses, is called a **balanced complete response structure**. Here a combination of a set of factors means the set of responses formed by the intersection of levels of each factor involved.

Given two factors A and B of a response structure, if the partition of responses generated by B is a refinement of the partition given by A, then factor B is said to be **nested** in A. More formally, factor B is said to be nested in A if each level of B appears with one, and only one, level of A. Throckmorton (1961) shows that, by considering the nesting relationship among the factors as a partial ordering relation on the set of responses, a balanced complete response structure can be represented by a lattice. A lattice is a partially ordered set in which every pair of elements has a least upper bound and a greatest lower bound. It can be shown that the set of factors of a response structure and the nesting relationship satisfy this definition.

A lattice structure can be represented visually by a diagram, called a **Hasse diagram**, by placing elements which cover other elements immediately above them in the diagram and connecting the elements and those elements which cover them with line segments. These are discussed extensively by Throckmorton (1961) and Kempthorne (1982). Thus balanced complete response structures can be represented by Hasse diagrams in which the nodes correspond to the factors and line segments denote nesting relationships. In the Hasse diagram of figure 1, μ represents the grand mean and E represents the error term. The factor E corresponds to the finest factor partition. All factors are nested in μ , and E is nested in all other factors. In addition, C is nested in A. If two factors are not connected by an upward link, they are **crossed**. Thus, B is crossed with A and C. The number in parentheses after each factor is the **range**, or number of levels. For a nested factor, say C, this number indicates that C has three levels for each level of A. Finally, fixed factors are underlined while

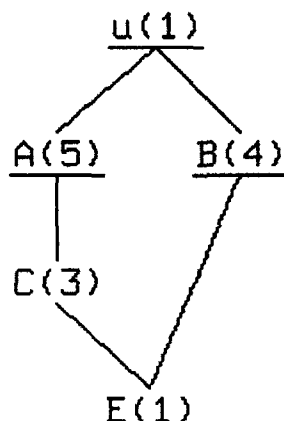


Figure 1: A Hasse diagram.

random factors are not.

3. Factor Sets

Factors sets are defined in terms of the nesting relationship and factor type (fixed or random). They play a central role in determining the ANOVA table and expected mean squares, and are easily obtained with reference to the Hasse diagram. We call the set of all factors in an experimental design the design set (DS), and define an effect as any single factor or any combination of 2 or more crossed factors. Every effect defines a different partition of the design set. The breakdown of the design set into factor sets is shown in table 1. The following relationships among the factor sets should be noted:

$$\begin{aligned}
 DS &= SF \cup CF \\
 SF &= LF \cup DF \\
 CF &= RCF \cup FCF \\
 RCF &= SRCF \cup NRCF
 \end{aligned}$$

Two numerical quantities associated with each effect and required for the ANOVA table are obtained from the factor sets. First we define the range of a factor as its number of levels and the diminished range as one less than the range. The symbolic product of an effect is computed as the product of the diminished ranges of its live factors and the ranges of its dead factors. The symbolic product generates the degrees of freedom for the given effect. The complement product of an effect is the product of the diminished ranges of its complement factors and generates the expected mean square coefficient for the given effect.

The final ingredient is the notion of formal interaction. The formal interaction of two effects is found by first juxtaposing all factors in the effects and then removing duplicates and factors that nest any other factors. The result is always an allowable effect. We now describe our system, beginning with an overview of LISP-STAT objects.

4. Representation

The LISP-STAT environment provides an excellent platform for object-oriented programming and customized dynamic graphics. The symbolic processing power of LISP is also available. We make use of these features to manipulate factor sets and to draw the structure diagram.

Object-oriented programming involves the use of computing entities, called **objects**, that contain both data and the procedures for processing that data. The familiar procedure call of other languages is accomplished by sending a message to an object to execute one of its methods. For example, if `reg-model` is a LISP-STAT regression-model object, we could ask it to plot its residuals with the expression

```
(send reg-model :plot-residuals)
```

In the LISP-STAT object system, there exist certain objects called **prototypes**. A prototype object contains the blueprint for creating new objects, called **instances**. Each instance has its own slots to hold data, but looks to its prototype for the code to execute methods. However, an instance may also have its own methods that either add to, or override, the prototype methods. Thus, code can be shared by many objects. In addition, prototypes may be arranged in an **inheritance hierarchy**, that allows prototypes to inherit, i.e. reuse, code from other prototypes.

Three new prototype objects are now introduced: `balanced-anova-proto`, `factor-proto`, and `structure-diagram-proto`. The first two prototypes inherit from the LISP-STAT root object `*object*`, and `structure-diagram-proto` inherits from the LISP-STAT prototype `graph-window-proto`. The main object of interest here is `balanced-anova-proto`; one of its slots will contain a list of factor objects (created from `factor-proto`) and another will contain a pointer to an instance of `structure-diagram-proto`.

4.1. Input

The balanced experimental design is implemented as `balanced-anova-proto`. Thus it contains slots to hold data and routines to process that data. The user begins by calling the function `balanced-anova-model`:

Symbolic Factors (SF)	Live Factors: factors that appear in the effect name (LF).	
	Dead Factors: factors that nest the live factors (DF).	
Complement Factors (CF)	Random Complement Factors (RCF)	Simple RCF: not nested in any other RCFs (SRCF).
		Non-simple RCF (NRCF).
	Fixed Complement Factors (FCF).	

Table 1: Factor Sets. The partition becomes finer as one moves from left to right.

```
(def ba (balanced-anova-model))
```

which returns a new instance of `balanced-anova-proto` in the variable `ba`.

This function also prompts the user to select a design file. It is assumed that one or more of these files already exist. Design files are plain text files that might look like

```
(A 5 fixed nil)
(C 3 random (A))
(B 4 fixed nil)
(E 1 random (A C B))
```

This design file generated the Hasse diagram in figure 1.

There is one line for each factor in the design. Each factor is represented as a list of four elements: the factor name (or letter designation), number of levels, type (either fixed or random) and a list of all factors in which the given factor is nested. An empty list, `nil`, in the fourth position indicates that the given factor is not nested in any other factors. Characters may be entered in upper- or lowercase, but LISP-STAT converts them to uppercase.

Each factor is read into a new instance of `factor-proto`, and the slots `name`, `range`, `type` and `nested-in` are initialized to the values in the input list. These factor objects are gathered into a list and stored in a slot, `factors`, in the `balanced-anova` object. Next, a list of all allowable effects in the model is determined and stored in another slot, `effects`. Finally, a menu is installed that allows access to the structure diagram, ANOVA table, expected mean square formulas, and factor sets. The user may also load another design from the menu to replace the current design.

4.2. Menu selections

The following `balanced-anova-proto` methods correspond to items in the menu. Method names always begin with a colon (`:`) and are followed by an argument list, where `()` means that no arguments are required.

- `:load-design ()` loads a design file. This method executes automatically when the function `(balanced-anova-model)` is called.

- `:draw-structure ()` draws the structure diagram. Each node in the diagram corresponds to a factor in the design. The vertical positioning of the nodes is determined by the nesting relationships, but in general it is hard to compute horizontal positions that result in a nice-looking graph. Thus, the user is able to rearrange the nodes by dragging them to the left or right with a mouse. One special node is added to complete the diagram: the grand mean which has one level, is fixed and nests all other factors.

- `:linear-model ()` writes the list of effects as a linear model.

- `:show-table ()` displays the ANOVA table, which has columns for sources of variation, degrees of freedom, and mean square ratios. Each allowable effect produces a line in the ANOVA. The value for the degrees of freedom comes from the symbolic product, and the F-test formula is derived via the following procedure. Form two sets of effects, S_e and S_o , defined as

$$S_e = \{e | e \text{ is a formal interaction of } LF \text{ with an even-way } (0, 2, 4, \dots) \text{ interaction of factors in } SRCF\}$$

$$S_o = \{o | o \text{ is a formal interaction of } LF \text{ with an odd-way } (1, 3, 5, \dots) \text{ interaction of factors in } SRCF\}$$

All n -way, $n = (0, 1, 2, \dots)$, formal interactions of factors in S_{RCF} are computed at once using the `:allowable-effects` method, which when given the set, S_{RCF} , returns the list of effects, $S_e \cup S_o$. Then the even-way and odd-way interactions are extracted according as the number of factors in the effect is even or odd. Finally, the numerator of the F-test is the sum of the mean squares of the effects in S_e , and the denominator is similarly obtained from S_o .

The sets, S_e and S_o always have the same size, say m . If $m = 1$, we have an exact F-test; otherwise

the ratio represents an approximate test of the Satterthwaite type.

- `:ems-list ()` displays formulas for the expected mean square of each effect in the model. Expected mean squares are computed by first forming the set

$$S = \{s | s \text{ is a formal interaction of } LF \text{ with an } m\text{-way interaction of factors in } RCF, \\ m = (0, 1, 2, \dots)\}$$

This set contains the effects that appear in the expected mean square with coefficients given by the complement product. Thus for a given effect, Q ,

$$EMS(Q) = \sum_{s \in S} k(s) \sigma_s^2$$

where $k(s)$ is the complement product of the effect, s , and σ_s^2 is a variance component if s is random, or a mean squared deviation from the treatment mean if s is fixed. An effect is random if any one of its live factors, LF , is random.

- `:factor-sets ()` displays the important factor sets (SF , LF , DF , CF , RCF , and $SRCF$) for a user-selected subset of effects.

4.3. Other methods

The following balanced-anova-proto methods are used by the methods described above and are available for investigating the balanced-anova object.

- `:factors (#optional factors)` returns the list of factors stored in the `factors` slot. If an argument is supplied, it will become the new slot value. The argument should be a list whose items are instances of `factor-proto` (see section 4.4).
- `:effects (#optional effects)` returns the list of effects stored in the `effects` slot. If an argument is supplied, it will become the new slot value. This slot can be reset with the following expression:

```
(send ba :effects
  (send ba :allowable-effects
    (send ba :factors)))
```

- `:allowable-effects (factor-list)` computes all allowable effects that can be derived from the list of factors and their nesting relationships. An effect is allowable if no factor in the effect is nested in another factor in the effect, i.e., if all of the factors in the effect are crossed with each other.

- `:complement-factors (effect)` returns the set of complement factors for effect.

- `:complement-prod (effect)` returns the complement product of effect.

- `:dead-factors (effect)` returns the set of dead factors for effect.

- `:effect-string (effect &key extended)` returns a string that writes the effect in the usual notation, e.g. B^*C . if `:extended` is `t` then the factors in which the effect is nested are included in parentheses, e.g. $B^*C(A)$.

- `:F-test (this-effect)` displays the F-test formula for this-effect.

- `:gfi (effect-list effect)` computes the formal interaction of effect with each item in effect-list and returns a list of the results. This method is used when computing the F-test and expected mean square formulas.

- `:MS-string (effect &key expected maxlen)` writes the effect name as a mean square name or expected mean square name (if `:expected` is `t`), e.g. $MS-B^*C$ or $EMS-B^*C$. the argument `maxlen` can be used to control the width of the printed text.

- `:random-CF (effect)` returns the set of random complement factors for effect.

- `:simple-RCF (effect)` returns the set of simple random complement factors for effect.

- `:symbolic-factors (effect)` returns the set of symbolic factors for effect.

- `:symbolic-product (effect)` returns the symbolic product of effect.

- `:write-ems (this-effect)` displays the expected mean square formula for this-effect, in which a variance component is denoted by the effect name in parentheses, and a mean squared deviation from a treatment mean is denoted by the effect name in square brackets. For example,

$$EMS-B = 15*[B] + 1*(C*B) + 1*(E)$$

since B is fixed, but $C*B$ and E are random effects.

4.4. Factor-*proto*

The prototype factor-*proto* is used to create factor objects. Each factor object contains the following slots, which can be accessed and changed using a method that matches the name of the slot, e.g., `:name`.

- `name` contains the name, or letter designation, of the factor.
- `range` contains the number of levels of the factor.
- `type` contains the symbol 'FIXED' or 'RANDOM'.
- `nested-in` contains a list of factor objects in which the factor is nested.
- `parents` contains a list of factor objects that are direct parents of the factor. See *structure-diagram-*proto** for more details.
- `x` contains the x-coordinate of the factor in the structure diagram. Used by *structure-diagram-*proto**.
- `y` contains the y-coordinate of the factor in the structure diagram. Used by *structure-diagram-*proto**.
- `depth` contains the depth of the factor in the structure diagram. Used by *structure-diagram-*proto**.

The only other method for factor-*proto* is `:print` which overrides the default method in the LISP-STAT object, `*object*`. The result of this is seen, for example, when evaluating the expression `(send ba :factors)`.

4.5. Structure-diagram-*proto*

The structure-diagram prototype contains the methods for drawing the structure diagram. When the `:draw-structure` message is sent to a balanced-anova object, the following actions take place:

- A new structure-diagram instance is created. It receives the list of factors from the balanced-anova object and stores it in the slot, `nodes`.
- A root node is added to represent the grand mean. All other nodes will be nested in this node.
- For each factor, the direct parents are computed and stored in the `parents` slot of the factor.
- For each factor, the depth is computed and stored in the `depth` slot of the factor. The grand mean has depth zero; its children have depth one, and so on.

- An initial location, (x, y) , is computed for each factor and stored in the `x` and `y` slots of the factor.
- The diagram is drawn in a LISP-STAT graphics window, and its memory address is stored in the *structure-diagram* slot of the balanced-anova object.

The user is now able to drag nodes horizontally to produce a "cleaner" diagram. Given LISP-STAT's dynamic graphics tools, it was much easier to let the user rearrange the final shape of the diagram than try to compute it up front.

The program is available from the statlib archive by sending an e-mail message to

`statlib@lib.stat.cmu.edu`

that contains the line:

`send hasse from xlipstat`

The archive contains the program, several design files and instructions for getting started.

5. References

- Kempthorne, O. (1982). *Classificatory Data Structures and Associated Linear Models*, in *Essays in Honor of C. R. Rao*, G. Killianpur, P. R. Krishnaiah, J. K. Ghosh, eds. New York: North Holland, 397-410.
- Searle, S. R. (1971). *Linear Models*. New York: Wiley.
- Tanimoto, S. (1990). *The Elements of Artificial Intelligence: Using Common Lisp*. New York: Computer Science Press.
- Taylor, W. H. and Hilton, H. G. (1981). A Structure Diagram Symbolization for Analysis of Variance. *The American Statistician*, 35, 2, 85-93.
- Throckmorton, T. N. (1961). *Structures of Classification Data*, unpublished Ph.D. dissertation. Iowa State University, Dept. of Statistics.
- Tierney, L. (1990). *LISP-STAT: An Object-Oriented Environment for Statistical Computing and Dynamic Graphics*. New York: Wiley.

Graphical Methods for Manufacturing Yield Improvement

Andrew J. Black

Harris Semiconductor

Engineering Systems Group

MS62-24 P.O.B. 883

Melbourne, Fl. 32901

ajblack@ms16.mis.semi.harris.com

Abstract

The manufacturing of very-large-scale integrated circuits involves the generation of large amounts of data which may be useful to isolate and solve manufacturing problems. In this paper graphical methods are presented which provide a means to discover patterns in high-dimensional data and identify potential causes of manufacturing yield problems. Data values are extracted from a large relational database containing information obtained at a number of processing steps, which are performed on the product at different locations. The relationship between data organization and the ease of executing specific analyses is discussed.

Introduction

This paper describes a set of graphical methods which are part of a sequence of steps which can be used to isolate the cause of failure of integrated circuit (IC) chips. The process will be illustrated by two examples.

The use of a relational database is critical in the process due to the large amounts of data, and the need to join data from several parts of the manufacturing process in order to draw reasonable conclusions with respect to the source of a shift in product yield.

There are several data sources; first, the wafer fabrication process steps, where information on what equipment was used, who performed the operation, and possibly a measure of the result of the operation. At wafer acceptance test (WAT), electrical testing is performed on individual transistors, and then final testing is performed on the chips themselves, prior to shipment to the customer. The data sources are often different physical locations many miles apart. The original format of the data varies widely. Wafer fab data is usually obtained from factory control software, while the WAT and test data is obtained from automated test equipment.

The relational database provides a unified environment for data access. The data stored in the database may be used for a variety of purposes, including cost accounting, production scheduling or yield improvement.

The Effect of Data Organization on Analysis

The way that the data values are organized into columns in the relational tables can significantly effect the ease by which the data can be retrieved and used by commonly available data analysis software products.

The tables can be organized in two basic formats which will be referred to as Long-Thin, and Short-Wide.

An example of a Long-Thin table appears in Table 1.

Table 1:

Id_1	Id_2	...	Id_p	Name	Value
A1234	30	...	1	r_pbase	0.34128
A1234	30	...	1	r_nbase	0.67754
A1234	30	...	1	v_offset	0.34253

In this format, $p + 1$ columns are required to uniquely identify a row (Id_1 to Id_p and 'Name'). For a semiconductor manufacturing test data table, the identifiers would typically be Lot, Wafer, and Part (i.e. chip or die). 'Name' contains the data 'tag', which is the name of a parameter, and 'Value' is the location of the data itself. This format requires $p+2$ columns in the table.

The Short-Wide table format is illustrated in Table 2.

Table 2:

Id_1	..	Id_p	r_pbase	r_nbase	v_offset
A1234	1	1	0.34128	0.67754	0.34253
A1234	1	2	0.39760	0.51059	0.34253
A1234	1	3	0.44157	0.82183	0.41497

For this format, $k+p$ columns are required for the table when k data values are associated with each row of the table. All data values associated with a unit are stored in the same row, and specifying a unit identifier returns all data values for that unit.

The Advantages of the Long-Thin structure are:

- Easy to add new parameters to the tables without changing the number of columns.
- Data on sets of related parameters can be extracted using pattern matching in SQL statements.

The Disadvantages of the Long-Thin structure:

- Domain of the Parameter_name column is undefined. Errors in data loading programs could install incorrect parameter names.
- It is difficult to obtain data values which have a compound where restriction on the 'Name' attribute.

Advantages of Short-Wide format:

- Minimizes the size of the table. Table is third normal form.
- Allows indexing on columns which may be frequently accessed.

Disadvantages of Short-Wide format:

- Table must be redefined when a new column is added.
- Several SQL commands are required to combine data from multiple columns into a single output column.

Several factors point to the use of the Short-Wide table structure to facilitate data extraction for subsequent analysis. Many common analyses such as scatterplots require paired vectors as input. Most ANOVA and boxplot functions are designed to receive a vector of data and a vector of classification values. Both of these data structures are cumbersome to construct using SQL commands against Long-Thin tables. When Short-Wide tables are used such paired vectors can nearly always be obtained with a single SQL statement.

Yield Resolution using Graphical Methods

IC chips are subjected to a number of electrical tests to quantify their performance in relation to published specifications for the device type. The result of the tests dictate whether the individual chip is suitable for sale under the published specifications for that part. If a chip passes all tests, it is acceptable for sale.

When a yield degradation is discovered, it is important to identify the nature of the change in process yield. IC chips are built on wafers made of silicon or other material, and the wafers are collected into lots, which are generally processed together. Various types of yield changes can occur due to the complexity of the manufacturing process.

The following major types of yield changes are seen:

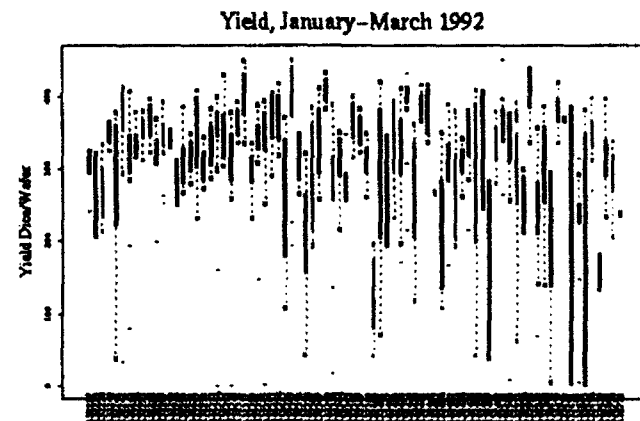
- An occasional isolated lot which exhibits high failure rate ('bad' lot).
- Clusters of bad lots.
- Major identifiable shift in the number of bad circuits for all lots on a particular process.

Within the lot the yield shift can be characterized by consistently elevated failure rates on all wafers in the lot, or by elevated failure rates occurring only on certain wafers within otherwise good lots.

A Case Study in the use of Graphics to Address Yield Fluctuation

A certain type of chip began to show unstable yields as determined by the count of saleable chips per wafer. The time-ordered boxplot of the distribution of the number of good circuits per wafer provides the capability to simultaneously determine the category of yield fluctuation at both the lot and wafer level. In Figure 1 it can be seen that lots which exhibit large amount of yield variability within the lot are interspersed with good lots. Each box represents a lot comprised of 15 to 30 wafers.

Figure 1



Once the general form of the shift in yield has been determined, the process of determining the source of the change is made easier by finding if there has been a shift in the relative rates of failure for the various electrical tests.

Similar electrical tests are commonly grouped into 'bins'. A pareto diagram of the count of bins can illustrate if a change in the relative failure rates has occurred for any of a set of 'bins'.

Figure 2 is a pareto diagram of the count of the number of

chips falling in each bin category for lots which exhibit a median yield of less than 250 dice/wafer or contain wafers yielding less than 200 dice/wafer (bad lots). Figure 3 shows the same data for the other lots from that period (good lots). Note the relative increase in the Bin 2 and Bin 3 circuits for the bad lots. For this circuit, it is known that Bin 2 and Bin 3 failures are often due to problems in the supply current of the device.

Figure 2

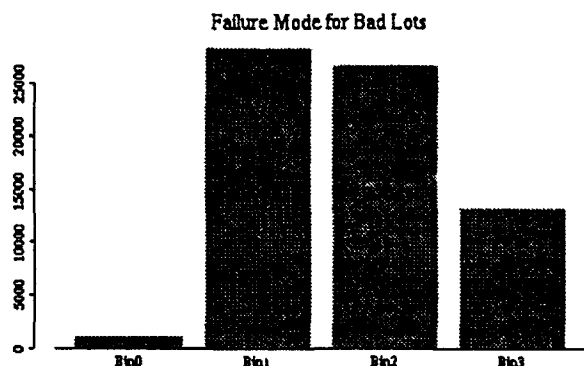
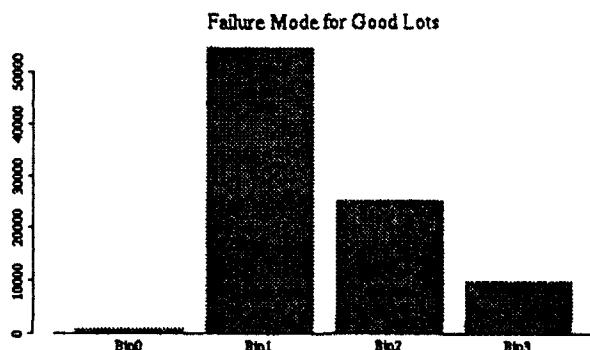


Figure 3



Since the kind of electrical test which has contributed to the yield fluctuation has been identified, it is beneficial to determine if differences in wafer acceptance test (WAT) parameters are detectable between the good and bad lots. WAT is testing which is performed on transistors similar to those in the chip, but which have been specially constructed so they may be tested individually.

One way to bring out these differences is to look at a 'standardized' boxplot [Bahrami, et.al., 1989]. This type of graphic allows the simultaneous comparison of several parameters on a unit-less scale which may contain a set of limits, usually the process specification limits. The boxplot is created by scaling the values in each distribution with respect to its specification limits. However, if the entire distribution is contained within the limits for most or all of the parameters, it is difficult to gain any useful information from

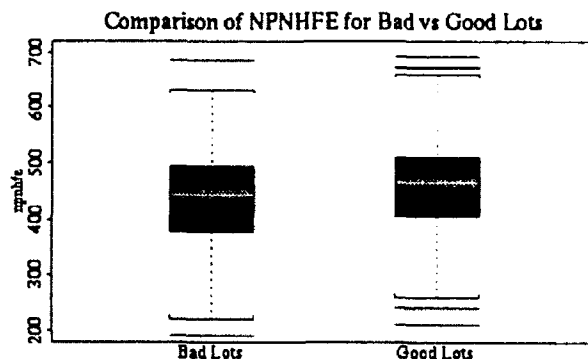
the plot. For this example, the standardized boxplot was not useful to identify a subset of the 35 WAT parameters which might have contributed to the yield fluctuation. Virtually all WAT test structures fell within 'acceptable' limits.

In this case, prior knowledge of the potential relationship between supply current failures and certain transistor parameters was used to create comparisons between good and bad lots.

The boxplot with confidence intervals [McGill, Tukey and Larsen, 1978] (Figure 4) illustrates a difference in npnhfe between good and bad lots. Similar differences were found for the pbase and nplus resistors, which are known to be related to npnhfe. No differences between the good and bad lots were observed for any of the other 30 WAT parameters. Two sample t-tests indicate a significant difference in mean npnhfe, pbase and nplus at the $p = 0.02$ level or less.

This result adds weight to the hypothesis that the likely cause of failure is associated with the npn transistor on the device. Since the only detectable difference in transistor parameters is in the npn device, the investigation will concentrate on process steps which are important for determining the performance of the npn transistor.

Figure 4



When clusters of sub-standard lots are observed, it is useful to answer the question of whether the poor-yielding lots have some segment of processing in common. If so, that particular processing step may bear closer scrutiny as the possible cause of the yield degradation. It would be desirable to be able to visually compare the process flow for selected lots, or to look for lots which were processed using the same equipment at a particular step.

If one wishes to test the hypothesis that a particular piece of equipment is responsible for causing the poor yield for the product, the distribution of yield for each of the parallel equipments at a processing step must be looked at individu-

ally. While this is potentially a laborious operation, considering there may be over one hundred steps, it is sometimes possible to narrow the search considerably.

For the current example, the base diffusion and the emitter diffusion are processing steps which are considered important for determining the characteristics of the transistor. Defects in the crystal structure of the silicon which are introduced during processing can cause supply current failures in the finished device. This can happen through the introduction of a contaminant during the emitter diffusion. At this step, two different diffusion tubes were valid for use during the time period in which the lots in question were processed. This is true for both the npn-type and pnp-type transistors. Since the chip contains both transistor types, and the steps are performed using different diffusion tubes, it is important to determine if either of the two steps has contributed to the yield failures.

The count of lots from each group (good and bad lots) which were processed using each of the two diffusion tubes was obtained from the wafer fab data table of the database. The data were entered into a 2X2 contingency table, and Fisher's exact test for independence between the categories was performed. The data for the npn transistor appears in Table 3. The null hypothesis of independence between the groups can be rejected with a two-sided p-value of about 0.02. Similar tests for the pnp transistor emitter diffusion showed a non-significant p-value of about 0.50. This substantiates the hypothesis that the source of the yield disturbance is the npn emitter diffusion.

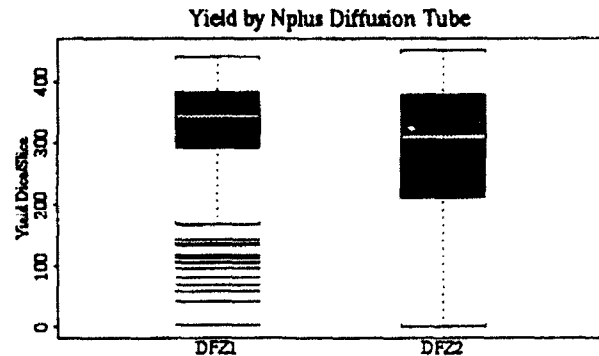
Table 3: NPN transistor N+ (emitter) diffusion

Category	DFZ1	DFZ2
Good Lots	17	4
Bad Lots	5	7

A comparative boxplot of the yield for the lots processed through each tube appears in Figure 5. While the confidence interval on the boxplot illustrates a questionable difference in medians at the $\alpha=0.05$ level, the t-test is significant at the 0.01 level.

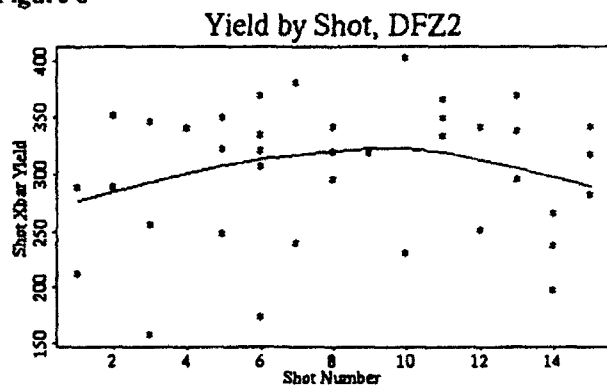
While not illustrated in this example, it may be useful in other cases to create similar comparative graphs to test for differences among shifts or operators who have handled the product instead of, or in addition to the tests for equipment.

Figure 5



It is useful to confirm that the defective lots were not processed in a group around the time of some event in the history of the equipment. Diffusion tubes are cleaned (or replaced) after a specified number of machine cycles. We can graphically illustrate the effect of the number of shots since cleaning has on the product yield. Figure 6 shows the lot average yield plotted against the shot number for an nplus diffusion tube. A lowess smooth curve [Cleveland, 1979] points out the trend in shot yield over the range of shot numbers. The best performance of the diffusion tube is in the center of the range, about 7 or 8 shots after preventative maintenance, with another 7 or 8 shots to go. There are no apparent anomalies which cluster the shots containing bad lots.

Figure 6



During the time period which was examined, it appears that the probability of defects being introduced in lots processed in DFZ2 was higher than those processed in DFZ1. It is interesting to note that both of the tubes were shut down for about two weeks over the holidays during the time period.

A Second Case Study

On a particular device type, occasional lots with *uniformly low yielding* wafers were observed. That is, the low number of total saleable dice in the lot was due to all the wafers in the lot having similarly low numbers of good chips. The type

of failure which was observed can often be associated with photolithography defects. Photolithography is the process by which chips are patterned by exposing photoresist through (photo)masks. Different instances of a mask are referred to as plates. A worker chooses one of several identical plates of the same mask to expose a lot on a high-volume product at a particular process step.

Since the defects are known to be possibly photolithography related, and occurred on all wafers of certain lots, a comparison of the yield across different plates was warranted. Contour graphs were created which illustrate the value of ICC (an electrical test) across the surface of the wafer for several mask plates which had been used to expose a particular level of the device. The graphs were obtained by calculating the median ICC for each x-y coordinate (chip location). The graph is therefore in the shape of a wafer, and represents a projection of the ICC data over several hundred wafers. The maximum value of ICC for a good chip is 0.12. Contour lines are drawn at 0.10, 0.15, 0.20 and 0.25.

Comparison of the contour plots illustrating the value of ICC for the sites on the wafer usually revealed a random pattern of failures for most mask plates (Figure 7). One plate showed a consistent pattern (Figure 8) of gross failures on the upper portion of the wafer. Later examination of that particular plate revealed a defect which had escaped the usual screening process, and had resulted in the loss of a considerable number of chips.

Figure 7

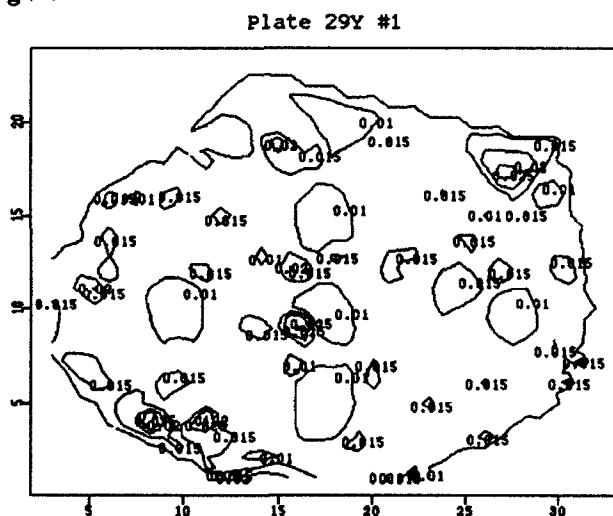
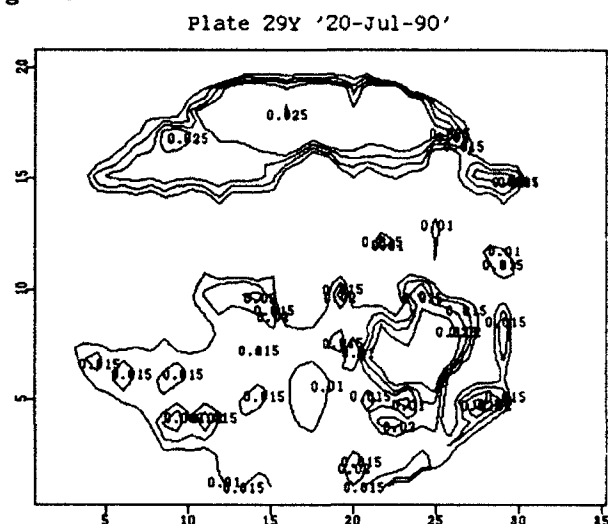


Figure 8



Conclusion

Simple graphical techniques, when coupled with easy access to relevant data in a relational format, can substantially cut the time required to isolate manufacturing yield changes. A graphical technique which allows the comparison of the process flow through various operations would be helpful to determine what pieces of equipment certain product groups have in common.

References

- Bahrami, M., et.al. (1989) A new Technique for Simultaneous Multiparameter Analysis, *Proceedings of the 21st Symposium on the Interface*, page 167
- Cleveland, W.S., (1979) Robust locally weighted regression and smoothing scatterplots. *Journal of the American Statistical Association* 74, 829-836.
- McGill, R. Tukey, J.W. and Larsen, W.A. (1978). Variations of box plots. *The American Statistician* 32, 12-16.

A COMBINED SHEWHART-CUMULATIVE SCORE PROCEDURE FOR PROCESS VARIABILITY

Matoteng M Ncube
Department of Mathematics and Statistics
University of West Florida
Pensacola, FL 32514-5751

ABSTRACT

Combined Shewhart-cumulative score (cuscore) quality control schemes are available for controlling the mean of a continuous production process. In many industrial applications, it is important to control the process variability as well. The proposed combined Shewhart-cumulative score (cuscore) procedure for detecting shifts in process variability uses the procedures developed by Ncube and Woodall (1984) to monitor shifts in the process mean of continuous production processes. It is shown, in the one-sided case, by average run length comparisons, that the proposed scheme performs significantly better than comparative Shewhart procedures and compares favourably to the standard cusum scheme.

INTRODUCTION

The problem of detecting shifts in the process variability has not received as much attention as that of detecting shifts in the process mean of continuous production processes even though it is important in the context of quality control.

Shewhart (1931) introduced the range chart (R-chart) to control variability of continuous production processes. At regular time intervals a sample of size $m \geq 2$ is taken and its range, $R_n = \text{largest observation} - \text{smallest observation}$ in the sample. Shewhart limits are appropriately set so that if any sample range falls outside these control limits an out-of-control signal would be indicated.

Page (1963) discusses the range Shewhart type quality control scheme with warning lines. In addition to the Shewhart control limits, Page suggested the addition of warning lines to the

control chart such that corrective action should be taken if either any sample range falls outside the action line or if any prespecified consecutive sample range values fall between the warning and action lines. Page (1963) also discussed range cumulative sum (cusum) scheme for controlling the standard deviation of normal populations. Sample ranges are cumulated continuously and corrective action will be taken as soon as the cumulation crosses some predetermined action line, namely the upper control limit (UCL) or the lower control limit (LCL). These procedures are more sensitive to small and moderate shifts in the standard deviation than the ordinary Shewhart type scheme.

Alt (1985) discusses Shewhart type procedures based on the sample variance, (s^2 -charts), the sample standard deviation (s-charts) and the multivariate cases for controlling process variability. In this case we cumulate the sample variances or the sample standard deviationns. An out-of-control condition is indicated as soon as the cumulation crosses a predetermined action line.

Hawkins (1980) and BS57003 part3 (1981) suggested using the transformation to variables that measure variability for cusum procedures to detect shifts in the standard deviation. In particular the transformed random variable $(S_n/\sigma_i)^{0.625}$ has been shown to be approximately normal for control purposes.

Ncube and Woodall (1984) proposed a combined Shewhart-cumulative score quality control scheme for detecting shifts in the process mean when the underlying process control variable is normal. A score of value $2h$, 1 , 0 or -1 is assigned to the sample mean less a

reference value depending on the pre-specified interval in which its value falls. These scores are then cumulated and an out-of-control condition is indicated as soon as the cumulation crosses a pre-assigned integer value, h , which is called the decision interval of the scheme. It is important to note that a cumulative score (cuscore) is a discretized cumulative sum (cusum). That is we discretize the sample statistic and then cumulate the scores. It has been shown that for detecting shifts in the process mean, the cumulative score procedures perform better than Shewhart schemes and compare favorably with cusum procedures. We hope that this better performance of the cuscore procedure over the Shewhart procedures for detecting shifts in the process mean will also be true for cuscore procedures for detecting shifts in process variability.

To this end we propose a combined Shewhart-cumulative score (cuscore) procedure for process variability that uses any one of the following statistics which measure process variability:

a) The sample range, R_n
 b) The sample variance, S_n^2 , or the sample standard deviation, S_n .
 c) The transformed variable, $T_n = (S_n/\sigma_t)^{0.625}$, $n = 1, 2, \dots$, where σ_t is the target standard deviation value and S_n is the sample standard deviation value. This random variable, for samples of size 3 through 20 from a normal distribution with mean, μ , and variance, σ^2 , has been shown to be sufficiently close to a normal distribution for control purposes, (BS57003 part 3 (1981)). We shall assume without loss of generality that $\sigma_t = 1$.

d) The transformation to a natural logarithm of the sample variance, $L_n = \ln(S_n^2)$, which is known to have an approximate normal random variable with:

$$\text{mean} = \mu_L = \ln \sigma^2 - \frac{1}{n-1} - \frac{1}{3(n-1)^2} + \frac{2}{15(n-1)^4}$$

and

$$\text{variance} = \sigma_L^2 = \frac{2}{L(n-1)} + \frac{2}{(n-1)^2} + \frac{4}{3(n-1)^3} - \frac{16}{15(n-1)^5}, \quad n = 2, 3, \dots$$

THE PROPOSED SCHEME

Suppose there is a continuous production process whose underlying process control variable is normal with mean 0 and variance σ_t^2 . At regular time intervals a sample of size $m \geq 2$ is taken and any one statistic which measures variability is calculated. Some commonly used statistics which measure process variability are: the sample range, the sample variance, the sample standard deviation, and the transformed random variables. We shall assume that these sample statistic values are obtained successively and are mutually independent.

In this paper we shall only discuss the one-sided case, assuming without loss of generality that we want to detect increases in process standard deviation. For the purposes of illustrating the proposed procedure we shall use the logarithm of the sample variance statistic, $L_n = \ln(S_n^2)$. Any of the other sample statistics: the sample range, R_n , the sample variance, S_n^2 , or the sample standard deviation, S_n could be used in place of L_n . The motivation for using the transformed random variable, L_n , is that its distribution is normal which is also the distribution of the target or parent population. For pre-assigned values K , k_1 , and k_2 a score, Y_n , is recorded as follows:

That is:

$$Y_n = \begin{cases} 2h, & k_2 \leq L_n - K < \infty, \\ 1, & k_1 < L_n - K < k_2, \\ 0, & -k_1 \leq L_n - K \leq k_1, \\ -1, & -\infty < L_n - K < -k_1, n=1, 2, \dots \end{cases}$$

where K is the reference value of the scheme. For detecting shifts in the process mean for cusum charts, Ewan and Kemp (1960) recommended a reference value to be halfway between the target

mean, value, μ_t , and a value of the process mean, μ_1 , that needs to be detected quickly. Similarly, for detecting shifts in the process standard deviation, using approximate normal random variables, the recommended value of the reference value will be halfway between the expected value of L_n at the target standard deviation value, σ_t , and at the value of the process standard deviation, σ_1 that needs to be detected quickly. Accordingly:

$$K = \frac{E(L_n|\sigma_t) + E(L_n|\sigma_1)}{2}$$
 For other non-normal random variables that measure process variability the recommended value of K is the geometric mean (square root) of the expected values of the random variable at the target standard deviation value, σ_t and at the value of the process standard deviation that needs to be detected quickly, σ_1 . Accordingly $K = \sqrt{E(L_n|\sigma_t)E(L_n|\sigma_1)}$. If we are taking samples of size five and we want to detect a 20 percent shift in the process standard deviation, a shift from $\sigma_t = 1$ to $\sigma_1 = 1.2$, the rule suggests taking K to be 0.1. For detecting a 50 percent shift in the process standard deviation, a shift from $\sigma_t = 1$ to $\sigma_1 = 1.5$, the rule suggests taking K to be 0.135155.

The values of k_1 and k_2 are usually chosen to be the number of standard deviations above or below the target standard deviation value.

Under the assumption that L_n , $n = 1, 2, \dots$, are independent normally distributed random variables with mean, μ_L , and standard deviation, σ_L , the random scores, Y_n , $n = 1, 2, \dots$, are independent and identically distributed with probability mass function given by:

$$P[Y = 2h] = P[L_n - K \geq k_2] = 1 - \phi\left(\frac{k_2 + K - \mu_L}{\sigma_L}\right) = s,$$

$$P[Y = 1] = P(k_1 < L_n - K < k_2),$$

$$= \phi\left(\frac{k_2 + K - \mu_L}{\sigma_L}\right) - \phi\left(\frac{k_1 + K - \mu_L}{\sigma_L}\right) = p,$$

$$P[Y = 0] = P(-k_1 \leq L_n - K \leq k_1),$$

$$= \phi\left(\frac{k_1 + K - \mu_L}{\sigma_L}\right) - \phi\left(\frac{-k_1 + K - \mu_L}{\sigma_L}\right) = r,$$

$$P[Y = -1] = P(T_n - K < -k_1) = \phi\left(\frac{-k_1 + K - \mu_L}{\sigma_L}\right),$$

$$= 1 - p - r - s = q$$

where $\phi(\cdot)$ is the cumulative standard normal distribution function. If the sample variance, S_n^2 or the sample standard deviation, S_n , were used in place of L_n , the values of s , p , r and q would have been obtained from the chi-square distribution function with $(n-1)$ degrees of freedom. If the studentized sample range, $W_n = \frac{R_n}{\sigma}$, is used instead of L_n , the values of p , q , r and s will be obtained from a chi-variate with appropriate degrees of freedom.

For the one-sided case, assuming without loss of generality that we want to detect increases in the process standard deviation we cumulate the scores, Y_n , in such a way that the cumulation, Z_n , is never allowed to be negative, and is restarted at zero whenever it becomes negative. Corrective action is taken as soon as the cumulation exceeds some preassigned value called the action line of the scheme.

That is: $Z_0 = 0$, and

$$Z_n = \min[h, Z_{n-1} + Y_n], \quad n = 1, 2, \dots$$

For this one-sided case, Z_n takes only the integer values $0, 1, 2, \dots, h$. This cuscore procedure is essentially a discretized cusum procedure. One way of analysing the properties of this scheme is to consider it to be a Markov chain since the current value of the cumulative score, Z_n only depends on the immediate past value of the cumulative score, Z_{n-1} . We shall say that the Markov chain, cuscore procedure, is in state i ($i < h$) whenever $Z_n = i$, $i = 0, 1, \dots, h-1$. As long as the Markov chain (Z_n) remains in these states, the process is said to be in-control. These states are called transient because they can communicate. When $Z_n \geq h$ the process is said to be in an out-of-control state or in an absorbing state. The transition probabilities of this Markov chain are given by:

$$p_{i,j} = \begin{cases} q+r, & i=0, j=0, \\ p, & j=i+1, i=0, 1, \dots, h-1, \\ q, & j=i-1, i=0, 1, \dots, h-1, \\ r, & j=i=0, 1, \dots, h-1, \\ s, & j=h, i=0, 1, \dots, h-1, \\ p+s, & j=h, i=h-1, \\ 1, & j=i=h, \\ 0, & \text{otherwise.} \end{cases}$$

For a Markov chain the transitional probability matrix, P can be written in the form:

$$P = \begin{bmatrix} R & p \\ 0' & 1 \end{bmatrix}$$

where for a cuscore procedure R is a tridiagonal matrix of transition probabilities from one transient state to another, p is a column vector of absorption probabilities, $0'$ is the zero vector.

Let N_i be the number of samples taken, starting from any transient state, i , before an out-of-control signal is indicated. N_i is called the run length of the process and the expected value of N_i (usually when $i = 0$) is called the average run length (ARL) of the process. That is: $ARL = \min[n : Z_n \geq h]$. The ARL is widely used as a means of comparing control charts and ideally we expect the ARL values to be high when the process is in-control and to drop sharply when it is out-of-control.

Brook and Evans (1972) showed that the ARL (average run length) of a scheme whose transition probability matrix is like P above can be obtained by adding all the elements in the first row of the matrix $(I - R)^{-1}$. $ARL(1)$ will denote the value of the average run length when the process standard deviation is at its target value, $\sigma_t = 1$ or when the process is in-control.

COMPARISONS

Performance comparisons of the different schemes will be based on the ARL. The in-control ARL of all the schemes will be set at the same high level of 200 and we will investigate the

values of the ARL when various shifts in the process standard deviation. The in-control ARL is the average run length of the scheme when the process standard deviation is at its target value, $\sigma_t = 0$. The scheme which has lower ARL values for a given range of shifts in the process standard deviation values is said to perform better than one with higher ARL values for the same range of shift values. Table I gives the average run length comparisons for the cuscore S-chart, the S^2 -chart, the $\ln(S^2)$, the T-chart and the Cusum S-chart. The cuscore $\ln(S^2)$ procedure performs uniformly better than the other four procedures for shifts in the process standard deviation considered. Table II gives the ARL values for the Shewhart S-chart, the cuscore S-chart in which for small to moderate shifts in the process standard deviation, the cuscore procedure performs better than the Shewhart procedures.

REFERENCES

- Brook, D. and Evans, D.A. (1972). An Approach to the Probability Distribution of CUSUM Run Length. *Biometrika*, 59, 539-549.
- Ewan, W.D. and Kemp, K.W. (1960). Sampling Inspection of Continuous Processes with no Autocorrelation Between Successive Results. *Biometrika*, 47, 363-380.
- Munford, A.G. (1980). A Control Chart Based on Cumulative Scores. *Applied Statistics*, 29, 252-258.
- Ncube, M.M. and Woodall, W.H. (1984). A Combined Shewhart-cumulative Score quality Control Chart. *Applied Statistics*, 33, 259-265.
- Page, E.S. (1954). Continuous Inspection Schemes. *Biometrika*, 41, 100-114.
- Shewhart, W.A. (1931). Economic Control of Quality of Manufactured Product. New York, D. Van Nostrand Co., Inc.

TABLE I

ARL COMPARISONS FOR THE CUSCORE
S-CHART, S^2 -CHART, $\ln(S^2)$ -CHART AND
CUSUM S-CHART

	Cuscore S-chart	Cuscore S^2 -chart	Curscore $\ln S^2$	Cusum S-chart
	$h=3$ $k=1.2$ $k_1=0.3$ $k_2=0.8$	$h=3$ $k=1.5$ $k_1=1.0$ $k_2=2.8$	$h=3$ $k=0.2$ $k_1=1.0$ $k_2=1.7$	$h=3.2$ $k=2.8$
1.0	200.0	200.0	200.0	200.0
1.1	51.4	56.6	31.6	46.0
1.2	20.6	22.9	10.5	19.0
1.3	11.2	12.3	5.4	10.0
1.4	7.3	7.8	3.3	7.1
1.5	5.3	5.6	2.3	5.3
2.0	2.2	2.2	1.1	2.5

TABLE II

ARL COMPARISONS FOR THE SHEWHART S-
CHARTS, AND THE CUSCORE-S-CHART

σ	Shewhart S-Chart $H=1.9275$	Modified Shewhart S-Chart $t=2$, $W=1.5798$ $H=1.907$	Cuscore S-Chart $H=3$ $k=1.151$ $k_1=0.2875$ $K_2=0.837$
1.0	200.00	200.00	200.00
1.1	65.04	59.93	51.40
1.2	28.27	25.07	20.60
1.3	15.04	13.20	11.20
1.4	9.25	8.16	7.30
1.5	6.32	5.66	5.30
1.6	4.67	4.25	4.20
1.7	3.66	3.39	3.40
1.8	3.01	2.83	2.90
1.9	2.56	2.44	2.50
2.0	2.24	2.16	2.20

Application of Experimental Design in Chemical Process Simulation

R. H. Wang and R. B. Safadi
Olin Research Center
350 Knotter drive
Cheshire, CT 06410

Abstract

Chemical process simulators are very useful in analyzing economic options, optimizing processes, and developing operating strategies. This paper illustrates the use of experimental design in model parameter estimation, process optimization and visualization of the effects of process variables on product characteristics. The use of orthogonal arrays can aid in the study of the effects of control and noise factors simultaneously with the minimal number of simulations. Examples are used to illustrate the principles of robust design in chemical manufacturing.

Introduction

Experimental design has been and is being used successfully in R&D labs and manufacturing locations throughout the world. Experimental design in these environments is used for several ends among which are the screening type designs, which are used to narrow down the number of factors to the ones that have the most impact on the desired effect(s), and the response surface type designs which fits a model of the desired effect(s) as a function of the various factors, which can be then used for optimization studies.

The above uses of experimental design can be employed in "computer" experiments in the form of simulation runs, and in particular, we have employed it in chemical process simulations. When one has a simulation program of a chemical process, experimental design can be used to build a simple model of it, estimate the model parameters, visualize the effects of the process variables that have the most impact, and optimize the process.

In this paper we will illustrate through the use of examples the benefits of the application of experimental design in chemical process simulation.

Process Simulation in the Chemical Industry

Simulation in the chemical process industries is found in steady state flowsheet simulation, reactor kinetic modeling, batch scheduling and simulation, and dynamics and process control. The use of simulation leads to a better understanding of the process, it saves on costly experimentation, enables extensive what-if studies, speeds up commercialization, and in general it can help minimize costs.

The stages in process simulation are:

- Model development
- Parameter estimation

- Model validation
- Case studies
- Process optimization

Experimental Design in Process Simulation

When one deals with simulations as opposed to actual laboratory experiments, the problem characteristics change somewhat. One typically deals with a large number of factors, strong nonlinear behavior, multiple responses, reproducible results, no missing data, and the need for randomization vanishes.

The available statistical designs are (Box et al., 1978):

- Factorial designs
- Fractional factorial designs
- Orthogonal arrays
- D-optimal designs
- Response surface designs
- Mixture designs
- Nested designs
- Superaturated designs.

The technical issues and the decisions that have to be made a priori are the scope of the study, the number of runs, selection of the particular design, optimization technique, data for model validation, visualization system, and the choice between a statistical model or a neural network based technique.

Examples

We will use two examples to illustrate different points. In the first, we will use a response surface design to fit a model to a styrene-maleic anhydride copolymerization reactor, then use this model to optimize the reactor. In the second, we will use a simple process synthesis example to illustrate the importance of using a simulation that includes the economics of the process in order to lead to meaningful conclusions.

Styrene-Maleic Anhydride Copolymerization

As seen in Figure 1, styrene and maleic anhydride and the initiator are fed to reactor 1, at a certain temperature T1, and the products, along with more of the feeds, are fed to reactor 2 at temperature T2.

The process variables are:

- Reactor #1
Styrene feed, S1
Maleic anhydride feed, M1
Initiator feed, I1
Reactor temperature, T1

- Reactor #2
Styrene feed, S2
Maleic anhydride feed, M2
Initiator feed, I2
Reactor temperature, T2

The response variables are:

- Conversions
Molecular weights
Product distributions

We generated a 50-run response surface design, and fitted the results into a full quadratic model. Figures 2-7 are a visualization of this model, where optimum values can be discerned.

EMPIRICAL MODEL - 1ST REACTOR/TEMP=70 DEG C

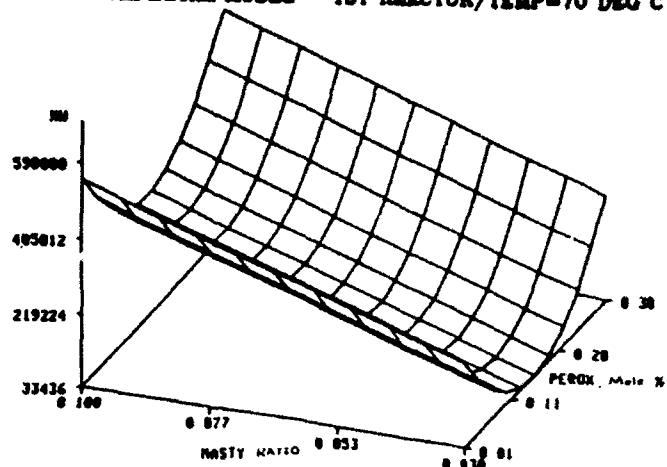


Figure 2.

Styrene - Maleic Anhydride
Copolymerization

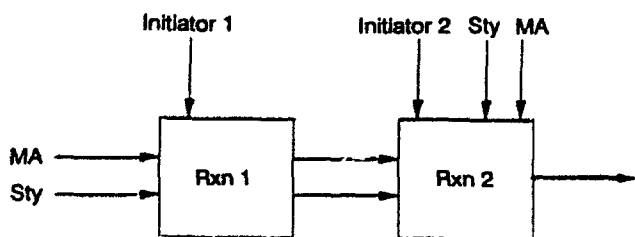


Figure 1.

EMPIRICAL MODEL - 1ST REACTOR/TEMP=70 DEG C

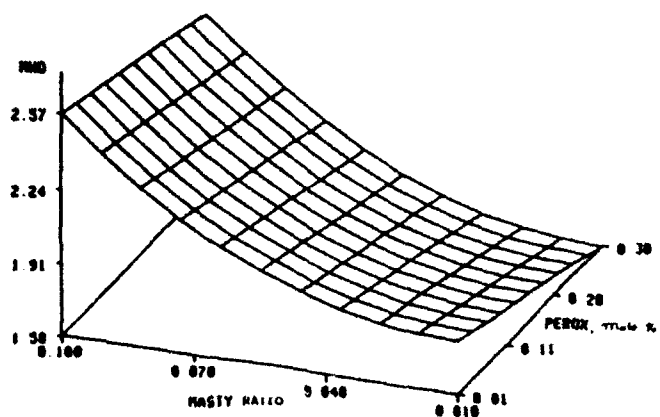


Figure 3.

EMPIRICAL MODEL - 1ST REACTOR/TEMP=70 DEG C

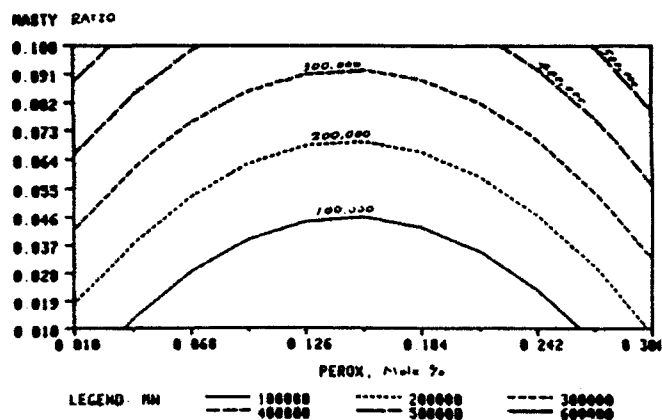


Figure 4.

EMPIRICAL MODEL - 1ST REACTOR/TEMP=70 DEG C

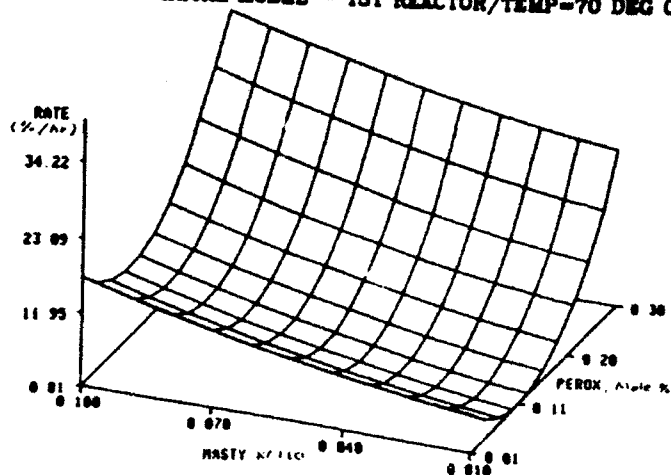


Figure 6.

EMPIRICAL MODEL - 1ST REACTOR/TEMP=70 DEG C

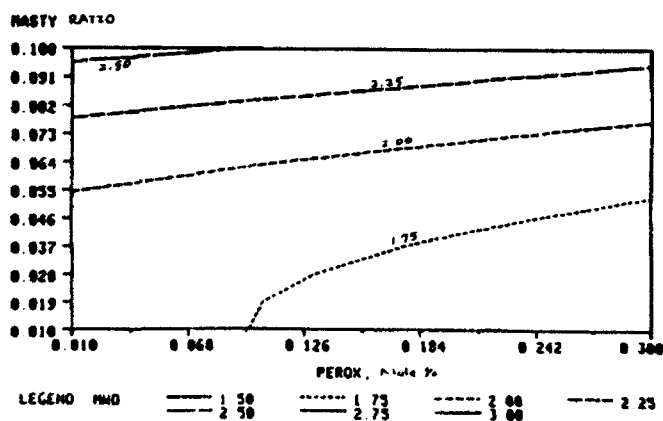


Figure 5.

EMPIRICAL MODEL - 1ST REACTOR/TEMP=70 DEG C

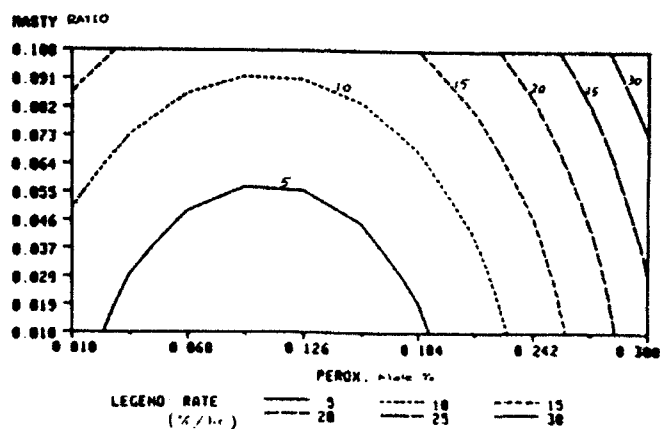


Figure 7.

Process Synthesis

Process synthesis is simply the conceptualization of a process that will realize on a commercial scale what had been accomplished in the lab. In Figure 8, we depict a simple process for the production of product B from feed A. B then reacts further to produce an undesired product C. The results of the chemical analysis in the lab system are shown in Figure 9, which shows an optimum yield to product B at about 95% conversion of A in the reactor. Once the economics of the plant are added, Figure 10 shows that the optimum conversion is actually a lot less, at about 65% (Safadi, 1990).

These results can be arrived at also when a signal-to-noise ratio study is performed (Phadke, 1989). Figure 11 shows that when the appropriate signal-to-noise ratio measure (Figure 12) is applied to both the yield and the plant profit, we arrive at two different optimal plant operating points. Obviously, the point that has the plant economics accounted for is the better choice.

Conclusions

Experimental design is very useful in parameter estimation, optimization, and visualization of chemical process simulations. Utilization of the design principle minimizes the number of simulations and improves the quality of the resulting informations. Robustness of products and processes can be greatly improved by using this approach.

References

1. Box, G., E., P., Hunter, W. G., Hunter J., S., Statistics for Experiments, John Wiley and Sons, New York, 1978.
2. Phadke, M. S., Quality Engineering Using Robust Design, Prentice Hall, Englewood Cliffs, NJ., 1989.
3. Safadi, R., B., Reactor System Synthesis and Design Based on Process Economics, Ph.D. dissertation, Chemical Engineering, University of Massachusetts, Amherst, MA, 1990.

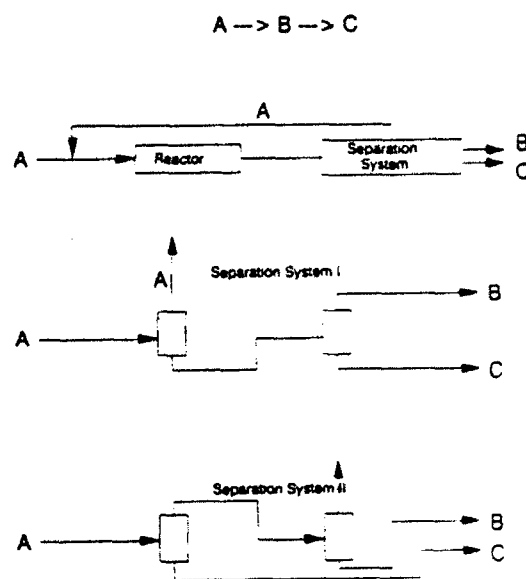


Figure 8.

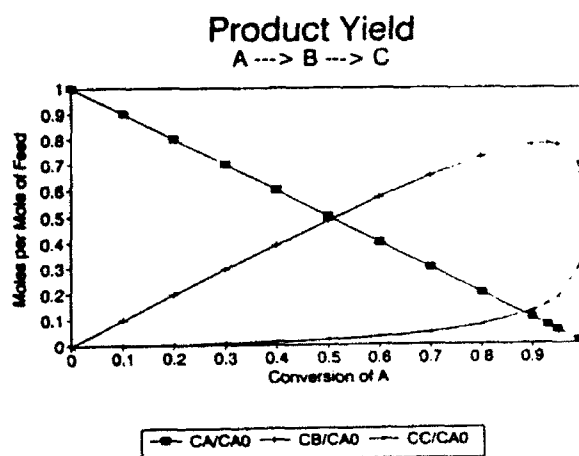


Figure 9.

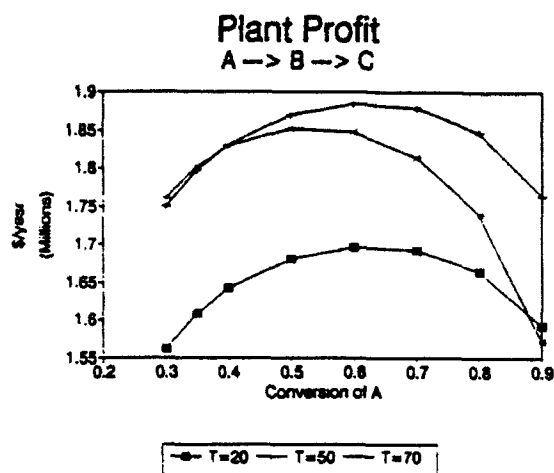


Figure 10.

Signal-to-Noise Ratio

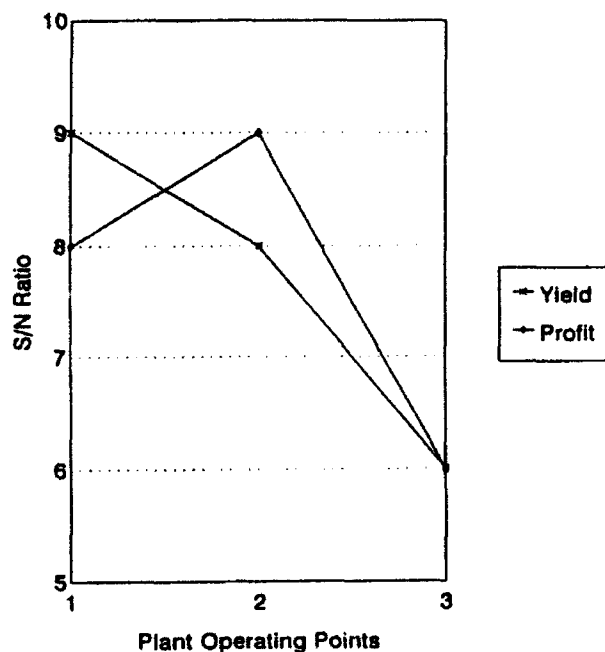


Figure 11.

Signal-to-Noise Ratios

If y_1, y_2, \dots, y_n represent values of a performance characteristic, then the Taguchi S/N ratios are as follows:

When the larger the performance characteristic the better.

$$S/N = -10 \cdot \log \left(\frac{1}{n} \sum \frac{1}{y_i^2} \right)$$

When the smaller the performance characteristic the better:

$$S/N = -10 \cdot \log \left(\frac{1}{n} \sum y_i^2 \right)$$

When a specific (nominal) value of the performance characteristic is best:

$$S/N = -10 \cdot \log \left(\frac{\bar{y}^2}{s^2} \right),$$

where

$$\bar{y} = \frac{1}{n} \sum y_i$$

and

$$s^2 = \frac{1}{n-1} \sum (y_i - \bar{y})^2$$

Figure 12.

Hierarchical Multivariate Visualization

T. Mihalisin and J. Schwegler

Temple University, Phila. PA 19122,
and Mihalisin Associates, Inc.
600 Honey Run Rd., Ambler, PA 19002

J. Timlin

Mihalisin Associates, Inc.
600 Honey Run Rd., Ambler, PA 19002

Abstract

A new patent pending method of plotting multivariate data is presented. This method allows one to see a coarse-grained or fine-grained representation of the n -dimensional data itself, not just projections of the data onto possible pairs of variables as is the case for methods that are based on scatter plot matrices. The new method involves the concepts of hierarchical axes which represent more than one variable by establishing a hierarchy of metrics for the variables and hierarchical symbols which represent the data in small n -dimensional cells, along lines of n -dimensional cells, planes of cells, etc. A variety of rules can be used to define these hierarchical symbols. Different rules are used for different types of multivariate visual data analysis such as fitting, conditional probability determination, significant mean differences, conditional distributions, factor and cluster analysis, etc. The technique allows one to visually analyze high dimensional data involving millions of bins and hundreds of millions of records in real time on a low cost workstation. The new method has significant performance and visual insight advantages over methods based on scatter plot matrices.

Introduction

For every data record taken as part of a carefully designed statistical study there are probably thousands if not millions of records that are gathered by business, government and various institutions for other purposes. This latter type of data is sometimes referred to as "serendipitous data". Since serendipitous data is not gathered in order to test a specific hypothesis or set of hypotheses a new type of data analysis, often called exploratory data analysis, is required. Exploratory data analysis makes extensive use of graphs to guide the user in the selection of appropriate analysis procedures. Applying statistics without viewing data plots is always a dangerous procedure even for data involving only a few variables and gathered as part of a seemingly well designed study. Large multivariate serendipitous data sets pose special problems which we address with new hierarchical multivariate visualization methods.

The problems associated with applying statistics to simple

X-Y paired data without viewing a plot of the data are well known¹. For that matter a histogram plot of a single variable's distribution is an essential first step for many types of analysis which may, for example, presume that all relevant variables have normal distributions or even that the normal distributions have the same standard deviation. A classic example of an analysis task that requires an X-Y plot is that of determining whether or not x and y are correlated. Blind evaluation of the linear correlation coefficient can lead to both false positive and false negative results¹ which may be difficult to detect in general even if one supplements the linear correlation coefficient r with other measures such as the number of sign changes for Δy (after sorting on x), or the mean distance of all the points from (\bar{x}, \bar{y}) where \bar{x} and \bar{y} are the mean values of x and y respectively etc. And yet the inappropriate nature of r , for a wide range of both false negative and false positive cases, can be made immediately obvious by simply viewing an X-Y plot of the data. This type of problem and other problems to be discussed below can of course be present in multivariate data sets as well. It is imperative that methods of plotting multivariate data be found in order to avoid the false positive and the false negative conclusions that can result from applying inappropriate statistical techniques to multivariate data sets.

In the simple two variable case a false positive for r can be obtained when for example all data points cluster around two points x_1, y_1 and x_2, y_2 . A false negative for r can be obtained if, for example, x and y are strongly correlated but only over a sub-interval of X . In the case of multivariate data the possibility of both clustering and/or conditional correlations are present. Consider Figure 1 which shows what appears to be uncorrelated X and Y variables both distributed more or less uniformly. If the data of Figure 1 correspond to the X and Y values of data points involving not two but say five variables e.g., u, v, w, x and y then Figure 1 would represent the projection of the 5 dimensional data onto the X - Y plane as is the case for example for one of the $5(4/2)=10$ distinct plots in the corresponding scatterplot matrix. It is entirely possible that if one constrained the values of one or more of the other variables i.e., u, v and/or w to lie in some narrow interval(s) then the corresponding subset of points in Figure 1 could show strong correlation. In dynamic scatter plot matrix type multivariate plotting² this constraining of one or more variables is

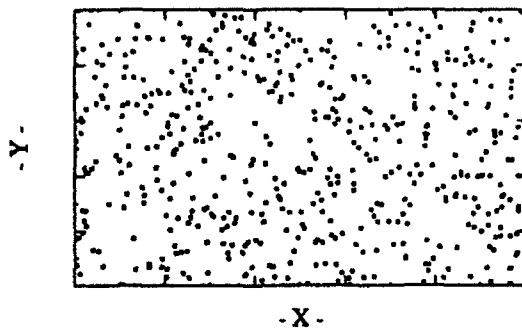


Figure 1 - X and Y values that appear to be uncorrelated when other variables are not constrained (see text).

done manually by the software user and is referred to as "brushing". The problem with this approach is that even for 5 variables the number of brushings required is large, and for 10 variables becomes astronomical. Consider for example the case where there are 5 variables u, v, w, x and y and one wishes to look for conditional correlation of any 2 of the 5. If one wishes to resolve each variable to 10% i.e., 10 equal intervals then there are $5 \times 10 = 50$ brushings corresponding to constraining just one variable, $5 \times 4 \times 10^2 / 2 = 1000$ corresponding to constraining two variables and $5 \times 4 \times 3 \times 10^3 / (3 \times 2) = 10,000$ corresponding to constraining three variables for a total of 11050 brushings. If one considers 10 variables instead of 5 and again one wishes to resolve each variable to 10% the number of ways of brushing 1 to 8 of the 10 variables in order to look for conditional correlation is 5,937,424,600. Clearly looking at over 5 billion brushed scatterplot matrices would be tedious even if the process were automated.

One of course is interested in a variety of visual tasks besides looking for conditional correlation of two variables. For example one may be interested in conditional probabilities or significant mean differences or factor analysis or cluster analysis etc. For the 10 variable case with each variable resolved to 10% there are generally on the order of 10^{10} statistical queries involving a single number e.g., conditional probability and of the order of 10^{20} statistical queries involving two numbers e.g., significant differences of means. Clearly if one is to attack the problem of visually analyzing large multivariate data sets new plotting techniques must be invented along with new interactive viewing tools.

The New Multivariate Plotting Method

The starting point for understanding our new method of plotting multivariate data on a 2d surface (i.e., paper or computer monitor) is to recognize that one may simply generalize

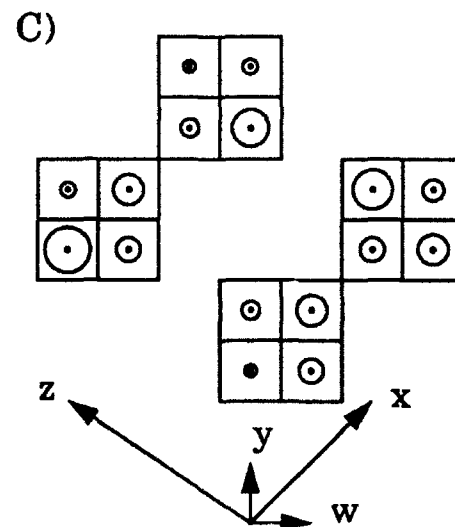
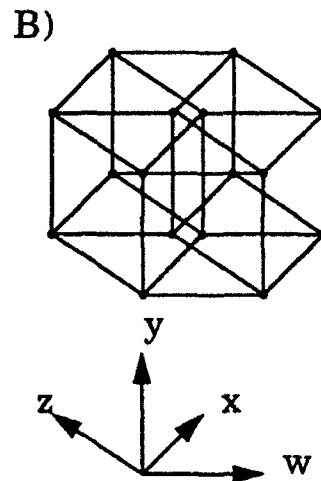
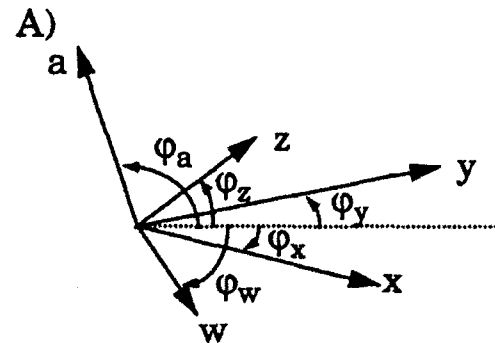


Figure 2 - Axes for 4 or more variables. A) vectors w, x, y, z and a for the corresponding variable axes can have any length and angular orientation. B) A 4d cube when w, x, y and z have compatible metrics. C) a 4d cube when w and y have smaller metrics than x and z .

the familiar method used for plotting 3 variables on a 2d display. For a starting orientation one normally picks X as horizontal, Y as vertical and Z at some intermediate angular orientation. Similarly for 4 or more variables one simply draws 4 or more vectors on the 2d surface as shown in Figure 2a. The fact that a particular display coordinate may correspond to many different points in the n dimensional space is not a new problem (for $n \geq 4$) but in fact is present even for $n=3$ i.e., conventional 3d graphics, and has led to a variety of techniques that help us cope with this multiplicity such as rotation of the coordinates or the dropping of normals from the points to say (the $Z=0$) X-Y plane. In Figure 2a the angles and the metrics (lengths) of the variable axes are to be considered as totally adjustable. Shown in Figure 2b are the 16 points corresponding to the vertices of a 4 cube (or to the sixteen 4d bins that would result when each variable in the 4 space is binned to 2 values each). In Figure 2b the metrics for variables V_i ($i=1$ to 4) or w, x, y and z are chosen to be of the same magnitude and all edges between each point and its 4 nearest neighbors are shown. Figure 2c on the other hand shows the same 16 points (and associated 4d bins represented as simple squares) for the case where the angles Φ_i ($i=1$ to 4) or Φ_w, Φ_x, Φ_y and Φ_z are identical to those of Figure 2b but differing metrics are chosen and edge lines have been omitted. Clearly Figure 2c is far less confusing and can easily be extended to cases where each variable is binned to higher resolution (more bins) and also to a higher number of variables while still remaining visually uncluttered and easy to understand. On the other hand extending the type of rendering shown in Figure 2b to either more bins/variable or more variables quickly results in a totally incomprehensible graph. Both the hierarchical metrics and the dropping of edge (or grid) lines in Figure 2c play an important role in producing its visual clarity. In previous articles we have considered important special cases that correspond to: 1) all vectors being colinear (say for example in the horizontal direction) but with hierarchical metrics (see Figure 3a) so that the entire "first" variable's binned axis (range) fits inside a single bin of the "second" variable and the entire "second" variable's binned axis (range) fits inside a single bin of the "third" variable etc. this special case is called the one hierarchical axis method^{3, 4, 5} and 2) all vectors being divided into 2 sets of hierarchical axes (see Figure 3b) (normally chosen to be at right angles to one another) this special case is called the two hierarchical axis method^{6, 7}. In this article we will also consider the case where all angles Φ_i are distinct but the metrics will differ from variable to variable. Figures 2c and 3c show examples of this type.

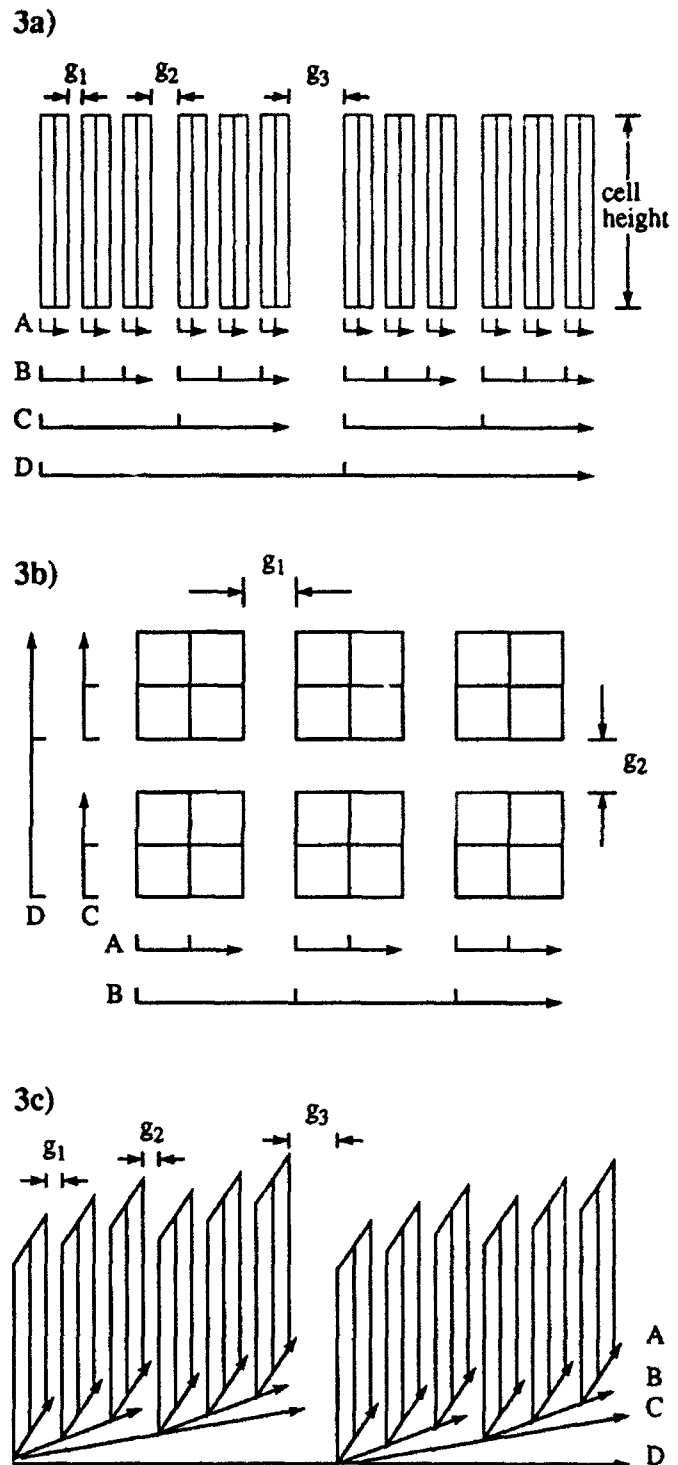


Figure 3 - Three ways to represent a space of 4 independent variable A, B, C and D which have been binned to 2, 3, 2 and 2 intervals respectively (see text). All of the gaps g_i are adjustable.

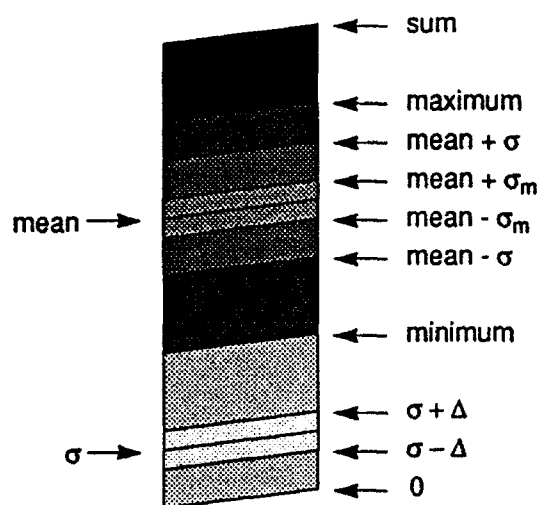


Figure 4a - Some important options for statistics which drive the heights of vertical parallelogram data symbols.

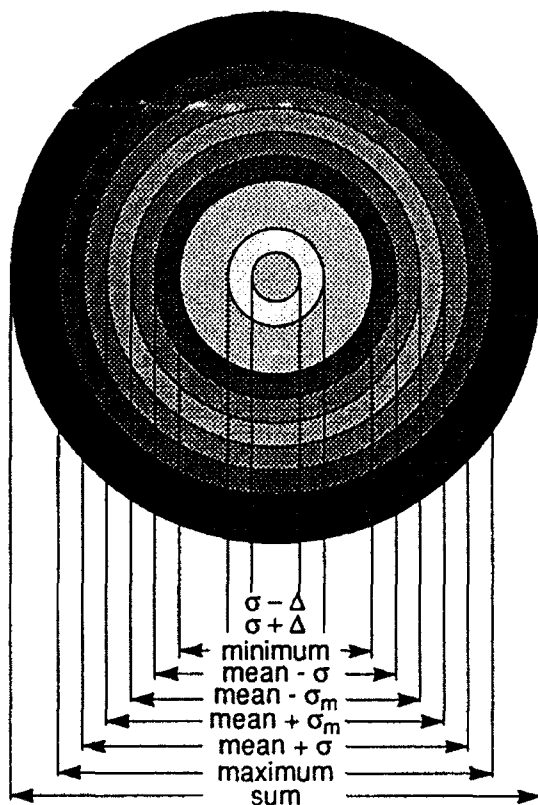


Figure 4b- Some important options for statistics which drive the radii or diameters of circular data symbols.

Symbols and Hierarchical Symbols

Note that inside of the cells shown in Figure 2c circles of varying radii are drawn as well as the 16 points that correspond to the vertices of the 4 cube. These circles can represent either the number of points in the corresponding 4d cell or the minimum or maximum or mean etc. of a fifth variable via either their radius or their area. Similarly each cell could contain multiple circles or any multiple attribute glyph and hence represent multiple dependent variables or multiple statistical attributes of one dependent variable e.g., maximum and minimum and mean etc. Similarly one could drive the radii of concentric circles and/or the heights of colinear parallelograms using a variety of statistics as shown in Figure 4a and Figure 4b.

Shown in Figure 5 is the case of 6 variables each binned to four values for a total of $4^6=4096$ six dimensional cells or bins. Note that instead of using circles or parallelograms the value of the dependent variable or the number of points in the cells is indicated by gray scale (dark being large). This transition eventually becomes necessary when the number of cells becomes very large and hence the area of each cell becomes too small to meaningfully drive the size of a symbol or symbols via either the number of points or the value of some other dependent variable or variables. Note however that if one zooms into a particular subspace of figure 5 for example the $t=1, w=1$ four dimensional subspace, or more precisely the 4 dimensional stack of 6 dimensional cells, then the representations of the cells can become larger and one can, for example, drive the radii of circles as shown in Figure 6. A complete set of tools are available that allow for viewing any subspace at any degree of granularity consistent with the total number of pixels.

If the total number of n dimensional bins exceeds the number of pixels then one must resort to hierarchical symbols representing data which has been summed or averaged etc. over two or more variables and/or degrees of granularity. For example Figure 6 could also represent the total number of points in u, t, v, w space with the two variables x and y having been summed over. Similarly Figure 5 could represent data with more than 6 variables in which case the smallest cells shown would represent sums or averages etc. over these variables, or Figure 6 could represent data involving 6 variables which are shown at a coarse granulation of 4 values per variable instead of at a fine resolution of 16 values per variable (since the latter would require $(4096)^2$ cells). In the latter case if one performed a "resolution zoom" into one "coarse" cell of Figure 5 one would obtain another figure similar in structure to Figure 5 (i.e. containing 4096 cells) but now each cell would correspond to a much smaller domain in the 6 dimensional space i.e. each coarse variable bin is sub-divided into 4 bins so that the 6 dimensional volume of the new finer cells is

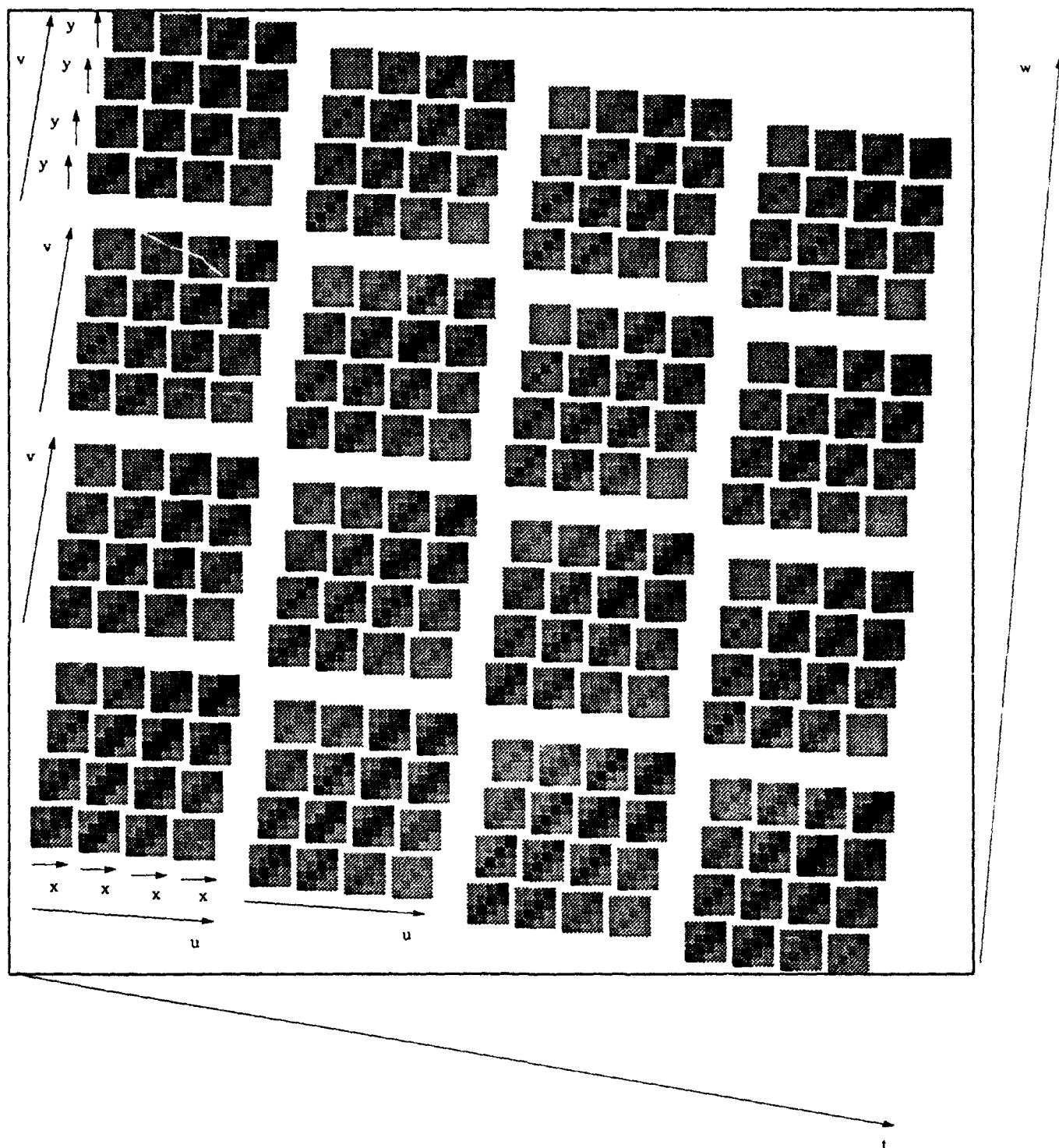


Figure 5 - A gray scale map for the number of points in a 6 dimensional space with each variable binned to 4 intervals. x and y have the smallest metrics followed by u and v. Finally t and w have the largest metrics. The correlation of x and y shows up as stripes running from the lower left to the upper right in the xy planes.

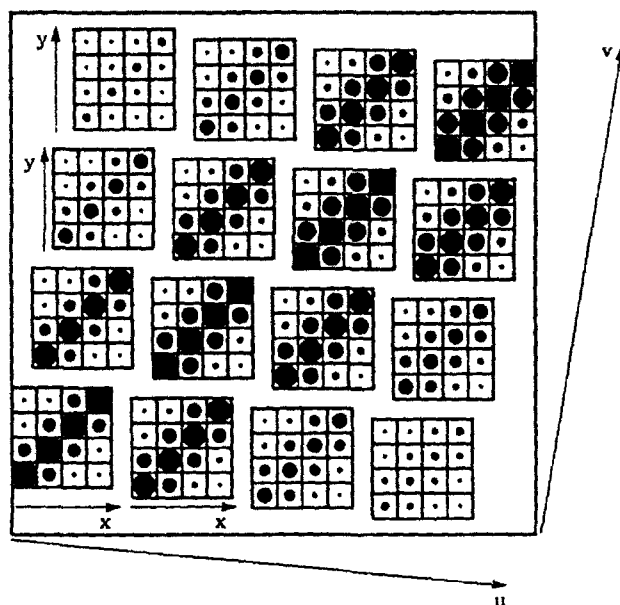


Figure 6 - A zoomed view of Figure 5 for $w=1$ and $t=1$. Here N is proportional to the radius (or area) of the circles.

1/4096 that of the original coarse cells.

Visual Analysis Tasks and Rules for Hierarchical Symbols

The rule or rules used to define the hierarchical symbols illustrated in Figures 4a and 4b are selected according to the type of visual analysis one wishes to perform. For example use of the sum rule and parallelograms is useful when considering all conditional distribution functions and all possible conditional probabilities. Figure 7a shows a plot of N the number of people versus age, education level and sex using the sum rule and a one hierarchical axis plot. The narrowest rectangles give the number of people versus age for each of the fixed education levels and for each sex. The intermediate width rectangles give the number of people versus education level for each sex (the ages are summed over). The two next widest rectangles give the number of males and females in the sample irrespective of ages and education levels (which have been summed over). And finally the one widest rectangle has a height driven by the total number of people in the sample. Hence several conditional distribution functions are visually represented. Moreover if one re-normalizes the height of any particular rectangle to unity then one can find the conditional probabilities represented by all narrower rectangles contained within it.

The min/max rule is useful for visual fitting. Figure 7b shows the function $w = e^{-x}e^{-y}e^{-z}$ again for a simple one hierarchical axis case. Note that the data are generated on a grid centered on the cells so that the narrowest rectangles have no vertical extent. That is there is only one point per cell hence $W_{\max} = W_{\min}$ in each of the smallest primitive cells.

The mean \pm the standard deviation of the mean is useful for locating significantly different means. Figure 7c shows the same census data used for Figure 7a again using a simple one hierarchical axis but now income is plotted on the vertical versus sex, education level and age. The narrowest rectangles show the mean income \pm the standard deviation of the mean for males and females versus education level for each of the age brackets. The second narrowest rectangles show mean income irrespective of (averaged over) sex versus education level for each age bracket. The next wider rectangles show mean income \pm the standard deviation of the mean irrespective of (averaged over) sex and education level versus age bracket. Finally the widest rectangle spanning the entire width of the Figure shows the global average for the sample \pm the standard deviation of the mean. In all there are 77 average incomes shown so that $77 \times 76/2 = 2926$ tests of significance can be made visually. To a first approximation two means are significantly different if they do not overlap in their vertical extents.

All Possible "Derived" Variables

One problem present for both the one and two hierarchical axis methods presented previously and for the current approach is that the hierarchy of metrics appears to imply that only certain sums or averages or minimums or maximums etc. over variables can be viewed for a given hierarchy of axes metrics. For example in our discussion of how Figure 6 involving 4 variables might relate to the parent data of Figure 5 involving 6 variables we remarked that a sum or average over the two variables with the smallest metrics x and y may have been performed on Figure 5 to obtain Figure 6. We could of course have chosen to say that a sum or an average over t and w or u and v etc. may have been performed but then no unique assignment of axis labels would have been possible. Similarly not all possible sums, min/max or means were shown in Figures 7a, 7b and 7c. A way to avoid these problems is shown in Figure 8 which demonstrates how one may generalize Figure 6 to obtain all possible derived variables e.g., sums on one graph. Basically one simply adds one value for each variable and displays in the corresponding "extra" cell the average or sum etc. over that variable. Consider the Σ_x circle indicated near the lower left corner of Figure 8 i.e. near $u=1, v=1$. There must be $4^3 = 64$ sums of the dependent variable over x corresponding to the 64 possible values of

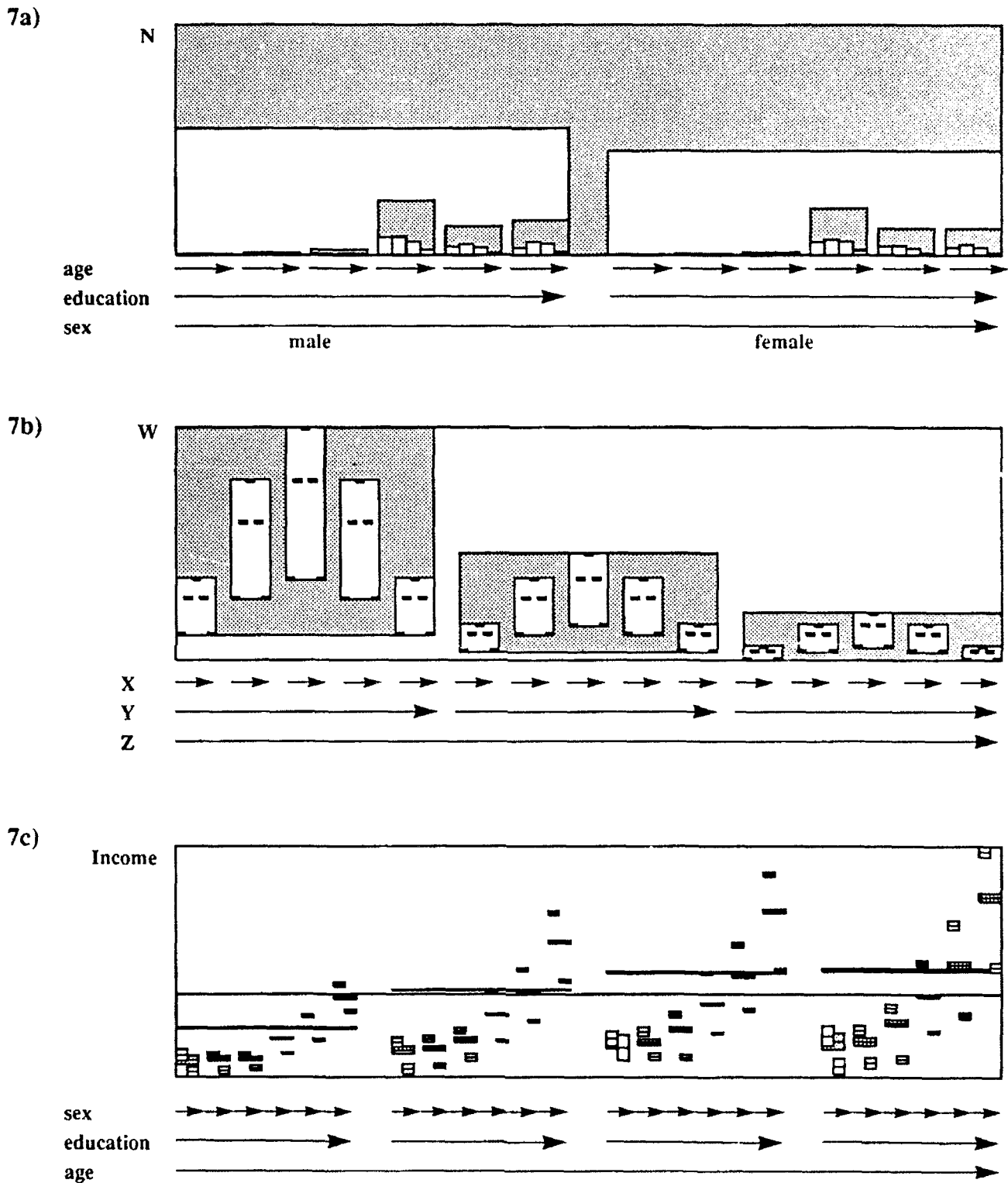


Figure 7 - Graphs illustrating three rules for determining the vertical sizes and/or locations of the hierarchical rectangles namely the sum rule, the minimum/maximum rule and the mean plus or minus the standard deviation of the mean rule for 7a, 7b and 7c respectively (see text).

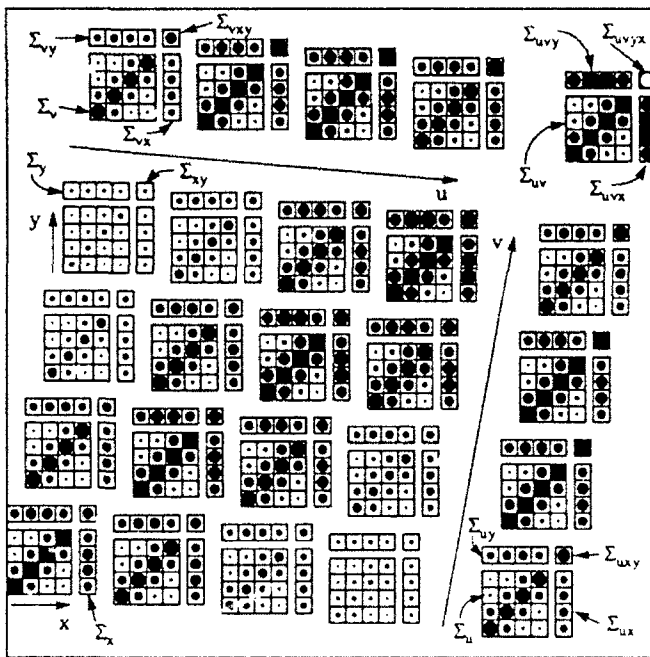


Figure 8 - By adding what appears to be a 5th value (offset from the other 4) for each variable x, y, u and v , one can show all possible sums as indicated. For each single sum e.g. Σ_x there are $4^3=64$ values, for each double sum $4^2=16$ etc. (see text).

(y, u, v) i.e. each variable x, y, u, v has been binned to four intervals. One can quickly find the other 3 circles representing Σ_x for $(u, v)=(1, 1)$. They are the 3 circles directly above the one in question. Hence the 4 circles vertically aligned for $(u, v)=(1, 1)$ represent the 4 values of Σ_x for $u=1, v=1, y=1$ to 4. There are 16 such vertically aligned groups of 4 circles corresponding to the 16 choices for (u, v) for a total of 64 values for Σ_x . Similarly there are $4^2 = 16$ values for double sums, 4 values for triple sums and of course only one quadruple sum. It should be noted that all cells corresponding to a single sum have the same scale. Similarly all cells corresponding to a double sum have the same scale (different from the single sum scale) etc. Similarly if one wished to show all possible derived variables originating from two operations e.g. summing and averaging one would add two "extra" values per variable.

All Possible Correlations

Since one is often interested in whether or not variables are correlated or conditionally correlated it is useful to plot all derived variables as just discussed but not for just one dependent variables but rather for all possible dependent variables.

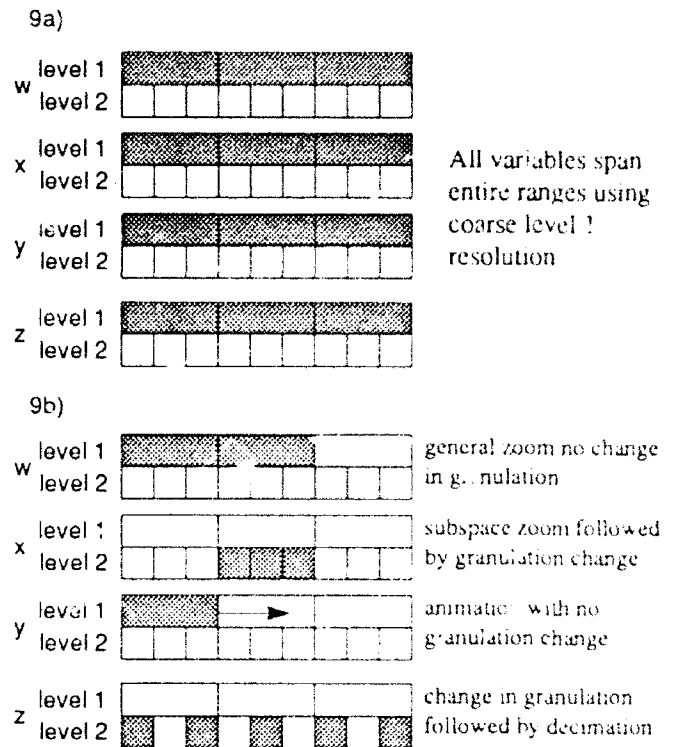


Figure 9 - Widgets used to display domain and resolution levels (See text.)

That is one treats all variables as the independent variables that are binned to form the space and also as dependent variables as well.

Tools, Graphically Guided Statistical/Database Queries and Granulation

Since the multivariate visualization system is capable of handling categorical as well as ordinal and continuous variables (at variable granulation levels) it is ideally suited for viewing large hierarchically organized databases that occur in the physical sciences and engineering as well as the social sciences, finance, actuarial sciences etc. A complete set of graphical widgets which represent the variables and their hierarchical granulation can be used to set any boolean algebra query or animate a set of queries. Shown in Figure 9a is a 4d space represented by 4 widgets with each variable "coarse" binned to three values. Shown in Figure 9b are the same widgets but here one has zoomed into the first two coarse values for w (level 1 granulation), the second for x (but its resolution has been increased to level 2), the third variable y is being animated at coarse level 1 and the last variable z has had its resolution increased but has been decimated to every

other bin. Hence as the animation proceeds through the three coarse level 1 values for x one sees a space corresponding to 2 coarse w bins, 3 fine x bins, and 5 fine z bins or $2 \times 3 \times 5 = 30$ cells for each frame of the animation.

Conclusions / Future Directions

By utilizing suitable subsets of the circle or parallelogram renderings shown in Figure 4a or Figure 4b, hierarchical granularity, and hierarchical symbols representing the data over lines or planes of cells etc. one can in fact perform a wide variety of visual data analysis on large multivariate data sets which is virtually impossible utilizing other multivariate visual analysis systems. In addition certain special cases for the angles and metrics shown in Figure 2a but not discussed here or in prior articles³⁻⁷ have very important applications particularly for data sets involving large numbers of variables or high dimensionality d where 2^d exceeds the number of pixels ($\sim 10^6$) i.e., for $d > 20$. These cases will be the subject of future articles. Finally it should be noted that although the primary application of this multivariate visual system is for large data sets wherein the numbers of points in the finest granularity cells are generally greater than one, the technique can also be useful for sparse data. For example the space of Figure 5 (6 dimensional) could be utilized to visualize a small number of points say 1 to 100 so that even at the coarse resolution level most of 4096 cells would be empty.

References

1. Chambers, John M., Cleveland, William S., Klierer, Beat and Tukey, Paul A. in "Graphical Methods for Data Analysis", Wadsworth International Group, Belmont, CA., 1983, pp75-82.
2. Becker, R.A., Cleveland, W.S., and Wilks, A.R., in "Dynamic Graphics for Statistics", eds. W.S. Cleveland and M.E. McGill, Wadworth and Brooks/Cole, Belmont, CA, 1988, pp1-12.
3. Mihalisin, T., Gawlinski, E., Timlin, J. and Schwegler, J., Scientific Computing and Automation Vol. 6, No. 1, Oct. 1989, pp.15-20.
4. Mihalisin, T., Suntech J., Vol. 3, No. 1, winter 1990, pp.25-31.
5. Mihalisin, T., Gawlinski, E., Timlin, J., and Schwegler, J., Proc. IEEE Conf. Visualization, San Francisco, Oct. 1990, pp. 255-262.
6. Mihalisin, T., Timlin, J., and Schwegler, J., IEEE Computer Graphics and Applications, Vol. 11, No. 3, May 1991, pp. 28-35.
7. Mihalisin, T., Timlin, J., and Schwegler, J., Proc. IEEE Conf. Visualization, San Diego, Oct 22-25, 1991, pp. 171-178.

GRAPHICAL METHODS FOR PERMUTATION DATA

Georgia Lee Thompson
 Department of Statistics
 Southern Methodist University
 Dallas, Texas 75275

Abstract

Ranked data occur when a group of "judges" are asked to either fully or partially rank a set of n items in order of "preference". The frequencies with which each ranking is chosen cannot be satisfactorily displayed with bar graphs or histograms because permutations do not have a linear ordering. However, both full and partial rankings of n items have a partial ordering that can be represented in a natural way on *generalized permutation polytopes* in \mathbb{R}^{n-1} . This paper illustrates the graphing of ranked data on generalized permutation polytopes [Thompson (1992)], and discusses solutions for the multivariate visualization problems encountered when graphing ranked data on higher dimensional polytopes.

1 Introduction

Fully ranked data occur, for example, when judges are asked to rank n items in order of preference. Each observation is a permutation of the first n integers, and the resulting set of frequencies is a function on the group of $n!$ permutations. In partially ranked data, the judges give a partial ranking of n items. For example, they may be asked to specify their first, second, and last choices. Each observation can then be written as a permutation of n not necessarily distinct numbers, and a set of partial rankings is a function on the set of all possible distinct permutations of the n not necessarily distinct numbers.

Although the group of permutations do not have a linear ordering, they do, however, have a natural partial ordering. It can be represented geometrically by the location of $n!$ vertices of a permutation polytopes. A polytope is the convex hull of a finite set of points in \mathbb{R}^n , and a permutation polytope is the convex hull of the $n!$ points in \mathbb{R}^n whose coordinates are the permutations of n distinct numbers. Thus, to represent a set of fully ranked data, the frequencies with which the permutation are chosen can be displayed, not on a line as is done with histograms,

but on the vertices of a permutation polytope.

To accommodate partially ranked data, the definition of permutation polytope is generalized. A *generalized permutation polytope* is defined to be the convex hull of the points in \mathbb{R}^n whose coordinates are the distinct permutations of n not necessarily distinct numbers. Then, as with fully ranked data, the frequency with which each partial ranking is chosen is displayed at the appropriate vertex of the generalized permutation polytope.

The resulting graphical displays are especially useful as diagnostic tools because they are compatible with two commonly used metrics for modeling ranked data: Kendall's τ and Spearman's ρ . For fully ranked data, Kendall's τ is the minimum number of edges that must be traversed to get from one vertex of the permutation polytope to another. And Spearman's ρ is proportional to the straight line distance between vertices. For partially ranked data, the straight line distance between two vertices is proportional to the fixed vector extension of Spearman's ρ [cf. Critchlow (1985)]. And the minimum number of edges that must be traversed to get from one point to another on a generalized permutation polytope induces a new extension of Kendall's τ for partially ranked data.

Closely related to this is the observation by McCullagh (1990) that the $n!$ permutations of n distinct numbers lie on the surface of a sphere in \mathbb{R}^{n-1} in such a way as to be compatible with both Kendall's τ and Spearman's ρ . Straightforward calculations further show that generalized permutation polytopes are inscribed in spheres in $n-1$ dimensional subspaces of \mathbb{R}^n . Hence, generalized permutation polytopes for displaying rankings of 3 and 4 items are inscribed in circles and spheres, respectively.

If more than four things are ranked together, then the problem of visualizing the polytope in higher dimensions must be addressed. One approach to this problem is to explore a higher dimensional polytope by examining its two, three, and four dimensional faces. In particular, for full rankings, all of the two-dimensional faces are combinatorially equivalent to either squares or hexagons, and all three dimensional faces are combinatorially equivalent to either

truncated octahedrons, cubes, or hexagonal prisms. If the data is partially ranked, then the two dimensional faces can also be triangles, and the three dimensional faces can also be equivalent to cubeoctahedrons, tetrahedrons, truncated tetrahedrons, octahedrons, and triangular prisms. Theorems that can be used to characterize all of the four and higher dimensional faces are found in Thompson (1992).

Other existing graphical methods for ranked data include multidimensional scaling, minimal spanning trees, and nearest neighbor graphs which are discussed by Diaconis (1988) for fully ranked data and by Critchlow (1985) for partially ranked data. Cohen and Mallows (1980) propose graphical methods based on multidimensional scaling and biplots. Cohen (1990) presents alternate exploratory data techniques for ranked data. And Baba (1986, 1988) proposes plots for ranked data that yield tests for concordance.

In Section 2, the graphical methods proposed by Thompson (1992) are discussed in detail for fully and partially ranked data when $n=3$ and $n=4$. An example is provided. Section 3 illustrates the graphical techniques in higher dimensions with several examples, and addresses the multivariate visualization issues.

2 Graphical Methods For $n=3,4$

Before discussing the proposed graphics for $n>4$, we will first focus on $n=3$ and $n=4$. Ranked data can be recorded either as an ordering or as a ranking. We will label items with letters, and orderings are denoted by permutations of the first n letters, bracketed by $\langle \rangle$. For example, $\langle b,c,a,d \rangle$ means that item b is ranked first, and item d last. A ranking is a permutation of n numbers written as a row vector $\pi = (\pi_1, \dots, \pi_n)$ where π_i is the rank of the item whose label is the i^{th} letter of the alphabet. The ranking corresponding to $\langle b,c,a,d \rangle$ is $(3,1,2,4)$.

Figure 1 shows the orderings and rankings of the 6 permutations of 3 items. Two adjacent points are connected by an edge if their orderings differ by a pairwise adjacent transposition, or equivalently, if their rankings differ by the inversion of two consecutive values. Hence, the minimum number of edges that must be traversed to get from one vertex to another is equal to Kendall's τ . Formally, if π and σ are two full rankings of n items, then $\tau(\pi, \sigma)$ is the number of pairs (i,j) such that $\pi_i < \pi_j$ and $\sigma_i > \sigma_j$. This is equivalent to the minimum number of pairwise adjacent transpositions needed to change the

ordering corresponding to π into the ordering corresponding to σ . The placement of the vertices in Figure 1 is also compatible with Spearman's ρ which for any n is defined as

$$\rho(\pi, \sigma) = \left(\sum_{i=1}^n (\pi_i - \sigma_i)^2 \right)^{1/2}.$$

If the edges of the regular hexagon are all of length $\sqrt{2}$, then Spearman's ρ is the Euclidian distance between two vertices.

These ideas extend to $n=4$ by placing the 24 permutations on the vertices of a truncated octahedron, as shown in Figure 2 [McCullagh (1990)]. The truncated octahedron has 8 hexagonal faces and 6 square faces. As in Figure 1, τ is the minimum number of edges that must be traversed to get from one vertex to another, and ρ is the Euclidian distance between two vertices if each edge has length $\sqrt{2}$ [cf. Schulman (1979)]. On the truncated octahedron, the 4 vertices of a square have the same 2 items ranked in the first 2 positions and the other 2 items ranked in the last 2 positions. Similarly, the 6 vertices of a hexagon all have the same item ranked either first or last. The idea that each face has a "defining property" is fundamental in the proposed graphical methods for $n>4$.

The plotting of ranked data with $n=4$ on truncated octahedrons is illustrated by the following example. At the start of a literary criticism course, 38 students read a short story and ranked 4 different styles of literary criticism in order of preference. At the end of the course, they read another short story and again ranked the same four styles of literary criticism. The 4 styles were authorial (a), comparative (c), personal (p), and textual (t); and one question of interest was whether or not the post-course rankings had moved in the direction of the teacher's own preferred ordering $\langle p,c,a,t \rangle$ [see Critchlow and Verducci (1989)]. The frequencies of the 38 pre-course rankings are shown in Figure 3a and the 38 post-course rankings are shown in Figure 3b. Most obviously, the frequencies do change a great deal between the two sets of rankings. First, there is an increase in the frequencies at the 6 vertices that correspond to orderings that begin with c . The post-course ranking do not seem to have moved toward the teacher's preferred ranking, $\langle p,c,a,t \rangle$, but they do appear to be closer to $\langle p,c,a,t \rangle$ than are the pre-course rankings. The orderings seem to have moved toward $\langle c,p,t,a \rangle$. McCullagh and Ye (1990) illustrate a similar conclusion by plotting the vectors of the average pre- and post-course ranking on a

truncated octahedron. Other observations are 1) the frequencies at the 6 vertices corresponding to the ordering ending in (c) decrease; 2) style (a) is rarely chosen as either a first or second choice after the course is completed; and 3) the incidence of style (t) as a first choice decreases.

Because the truncated tetrahedron is inscribed in a sphere, it can be mapped onto the plane via a central projection from any point of the sphere. The result is a Schlegel diagram as described by Banchoff (1990). This mapping can also be described as "puncturing" one of the two dimensional faces and then stretching the resulting hole out big enough until the polytope can lie flat in a plane. The stretching and distortion that occur around the edges can often be minimized by puncturing a face that is as far away from the bulk of the data as possible. The pre- and post-course rankings of the literary criticism styles (cf. Figures 3a and 3b) are plotted on projected truncated octahedrons in Figures 4a and 4b. In these figures the hexagon containing all orderings that end with c has been punctured. Most of the same features of the data seen in Figures 3a and 3b are evident in Figures 4a and 4b. To enhance the visual content of the graphs, the stretching is not uniform around the figure. However, all convex faces are still convex. Also, if a square face had been punctured instead of a hexagonal face, the resulting projection would look quite different.

To make the plots perceptually accurate, the areas of the circles in Figures 3a, 3b, 4a, and 4b are based on Steven's Law which says that the perceived scale, p , of the size of an area is

$$p \propto (\text{area})^{.7}$$

(Cleveland, 1985). Hence, the radii of the circles are calculated as

$$\text{radii} \propto f^{5/7},$$

where f is the frequency. If the areas are proportional to the values, i.e., $\text{area} \propto f$, then small circles appear too large and large circles appear too small. Conversely, if the radius of the circle is proportional to the frequency, i.e., $\text{radius} \propto f$, then large values are magnified and small values are minimized.

Generalized permutation polytopes for partial rankings can be constructed from permutation polytopes for full rankings by simply pinching together the vertices corresponding to the full orderings that are not distinguished by the partial ordering. For example, if $n=3$ and the judges are asked to specify only their first choice, then the vertices of the hexagon in Figure 1 that correspond to the orderings $\langle a,b,c \rangle$ and $\langle a,c,b \rangle$ are pinched

together into one point. Similarly, the two orderings beginning with b are pinched into one point as are the two orderings beginning with c. The resulting figure is a triangle in which each vertex corresponds to one of the three possible first choices. If $n=3$ and the judges are asked to specify only their last choice, the resulting figure is again a triangle, but with each vertex corresponding to one of the three last choices.

The generalized permutation polytopes for partial rankings with $n=4$ can be determined similarly. If the first and second choices are specified, but no distinction is made between the last two choices, then the resulting figure is a truncated tetrahedron as shown in Figure 5a. Each of the four triangular faces corresponds to the partial rankings in which the same item is ranked first; each of the four hexagonal faces corresponds to the partial rankings in which the same item is ranked last. Similarly, if the third and fourth choices are specified, but no distinction is made between the first two, the figure is again a truncated tetrahedron. If just the first choice or just the last choice is specified, then the resulting polytope is a tetrahedron shown in Figure 5b. Next, suppose that both the first and last choices are specified, but no distinction is made between the second and third choices. Then the resulting polytope is the cubeoctahedron shown in Figure 5c. It has twelve vertices, six square faces, and eight triangular faces. And lastly, if two items are selected as first choices and two items are selected as last choices, then the resulting polytope is the octahedron shown in Figure 5d. Partially ranked data of 4 items is plotted on one of the above polytopes by placing at the appropriate vertices circles scaled according to Steven's Law.

3 Graphical Methods For $n > 4$

As discussed in detail in Thompson (1992), the ideas in Section 2 can be extended to higher dimensions. The generalized permutation polytope for a data set of either full or partial rankings of n items is inscribed in a sphere in \mathbb{R}^{n-1} . The vertices on which the frequencies are graphed lie on the surface of the sphere. On the higher dimensional polytopes, the metrics Spearman's ρ and Kendall's τ have the same properties as they do for $n=3$ and $n=4$. However, graphing data on the higher dimensional polytopes presents visualization problems.

One approach to the multivariate visualization problem is to determine all of the two, three, and four dimensional faces of the higher dimensional polytopes

[cf. Thompson (1992)]. Often, the data can be effectively illustrated by a sequence of three dimensional polytopes in which the frequencies are plotted on the appropriate vertices. Frequently, it is also useful to plot one or more of the 4 dimensional polytopes. One technique for plotting 4 dimensional polytopes is to use the fact that the four dimensional face is inscribed in a sphere in \mathbb{R}^4 . The surface of this sphere in \mathbb{R}^4 can be mapped into \mathbb{R}^3 just as the truncated octahedrons in Section 2 were mapped into the plane in Figures 4a and 4b. Frequently, this mapping can be chosen in such a way as to preserve some distance information and to minimize, in some sense, the distortion due to the stretching at the edges. In particular, the polytopes in \mathbb{R}^4 for full and partial orderings of 5 things can be mapped into three dimensions, and their projections drawn on a piece of paper.

To illustrate this idea for graphing rankings of 5 items, we consider the partial rankings found in the APA voting data [Diaconis (1988)]. The data set contains the results of an election in which each member of the American Psychological Association voted for the president of the association by ranking the 5 candidates. While many ballots contained full rankings of the 5 candidates, a number of ballots contained only partial rankings. Three different partial rankings were obtained: just a first choice; just a first and a second choice; and a first, second, and third choice. In this example, the candidates are labeled with the numbers 1 through 5 instead of with the letters a, b, c, d, and e.

Looking first at the ballots in which only a first place candidate was indicated, we see that the corresponding generalized permutation polytope has 5 vertices: one for each candidate. The 5 vertices, which lie on the surface of a sphere in \mathbb{R}^4 , are all connected to each other by a total of 10 edges. The polytope has 5 three dimensional faces, all of which are tetrahedrons, and 10 triangular two dimensional faces. If one of the tetrahedrons is punctured and stretched out, the polytope can be projected into \mathbb{R}^3 . Figure 6 contains the graph which results from puncturing the tetrahedron with vertices 1,2,4, and 5. The most obvious feature of the graph is that the 5 candidates were chosen almost uniformly, with candidates 3 and 5 being preferred somewhat more than candidates 1, 2, and 4.

The partial rankings containing just first and second choices for president can be graphed on a polytope with 20 vertices and 40 edges that is inscribed in a sphere in \mathbb{R}^4 . Each vertex is connected

to 4 other vertices. The polytope has 10 three dimensional faces: 5 tetrahedrons, each of which has the same candidate ranked first, and 5 truncated tetrahedrons, each of which has the same candidate ranked among the last three. One projection of the polytope into \mathbb{R}^3 is shown in Figure 7a with the data graphed on it. The polytope in Figure 7a can be drawn from the one in Figure 6 by replacing each of the 5 vertices with a tetrahedron. The tetrahedron in which candidate 2 is ranked first is shown off to the side; in the projection it is hidden behind the tetrahedron with candidate 3 ranked first. Immediately evident from Figure 7a is that the data is not uniform. Candidates 1 and 3 are chosen as a pair much more frequently than are any other pair of candidates. Candidates 4 and 5 are also chosen slightly more often than the remaining pairs of candidates.

The polytope in Figure 7a was obtained by puncturing and stretching the three dimensional truncated tetrahedron whose vertices correspond to the rankings in which candidate 3 is not a first or second choice. If, instead, a tetrahedron is punctured and stretched, the resulting projection into \mathbb{R}^3 looks quite different. Figure 7b shows the resulting projection with the data graphed on it. It is formed by puncturing the tetrahedron whose vertices correspond to candidate 1 always being first choice. The vertex $\langle 1,2 \rangle$ is shown to the right of the Figure 7b; in the projection it is hidden behind $\langle 2,1 \rangle$. The same features of the data seen in Figure 7a are also visible in Figure 7b.

Lastly, we graph the partial rankings that contain first, second and third choices. The generalized permutation polytope for this data, which is inscribed in a sphere in \mathbb{R}^4 , has 60 vertices and 90 edges. Each vertex is connected to three other vertices. It has 20 three dimensional faces: 5 truncated tetrahedrons, 5 truncated octahedrons, and 10 triangular prisms. In Figure 8 the data graphed on a projection of this polytope. The polytope is obtained from the polytope in Figure 7a by changing each tetrahedron into a truncated tetrahedron, and each line segment connecting tetrahedrons into triangular prisms. The truncated tetrahedron with vertices that have candidate 2 ranked first is shown to the side of the figure; in the projection, it is hidden behind the truncated tetrahedron in which all vertices have candidate 3 ranked first. Figure 8 shows that the data is not uniform. Points on the triangular prism in which all vertices start with candidates 3 and 1 ranked either first or second are chosen frequently.

There is also a cluster of data around the triangular prism in which candidates 4 and 5 are ranked in first and second place. That cluster seems more diffuse than the cluster of data around the pair 1 and 3.

In the next example we will consider a data set consisting of partial rankings of 11 items. In a survey a charitable organization asked potential contributors to rank from 1 to 3 the top three out of eleven needs in the community. The data is listed in Thompson (1992). Altogether, 576 respondents indicated their first, second, and third choices; and 284 of the possible 990 partial rankings were chosen.

The resulting generalized permutation polytope on which this data can be "graphed" is inscribed in a sphere in \mathbb{R}^{10} and has 990 vertices. Each vertex is connected to 10 other vertices. The analysis in Thompson (1992) shows that the three dimensional faces consist of 330 truncated octahedrons, and thousands of truncated tetrahedrons, tetrahedrons, and triangular prisms. Because of the large number of 3 dimensional faces, it is impractical to examine the data by looking at all or even most of the 3 dimensional faces. In this case, a more productive approach is to find one or more 4 dimensional faces that contain as much of the data as possible.

For example, if interest is restricted to only five items, then the data can be plotted on the same polytope as in Figure 9. This is done in Figure 9 for choices F, E, D, H, and P where F = food and financial assistance for families in crisis, E = employment assistance for the unemployed, D = day care for low income families, H = housing for low and moderate income families, and P = prepared meals and health services for low income senior citizens. The truncated octahedron corresponding to rankings beginning with P is hidden behind the figure, and is shown at the bottom of Figure 9. One striking feature the 4 dimensional face shown in Figure 9 is that it contains almost 1/3 of the data on only 20 of the thousands of possible 3 dimensional faces. Hence, a large portion of the data that is naturally represented on a 10 dimensional polytope has been captured by a single drawing in the plane. The truncated tetrahedron in the middle of Figure 9 in which F is ranked first contains 92 responses or 16% of the data. Interestingly, the points on this truncated tetrahedron are chosen almost uniformly except for FDP and FDH. The frequency distributions on the other 4 truncated tetrahedrons are all fairly similar to each other, but overall their frequencies are considerably less than those of the truncated tetrahedron beginning with F.

4 References

- Baba, Y. (1986). "Graphical Analysis of Ranked Data," *Behaviormetrika*, 19, 1-15.
- Baba, Y. (1988), "Graphical Analysis of Ranks," In E. Diday et. al. (eds.) *Recent Developments in Clustering and Data Analysis*, New York: Academic Press.
- Banchoff, T. F. (1990), *Beyond the Third Dimension*, New York: Scientific American Library.
- Cleveland, W. S. (1985), *The Elements of Graphing Data*, Monterey: Wadsworth Advanced Books and Software.
- Cohen, A. (1990), "Data Analysis of Full and Partial Rankings," Technical Report, Dept. of Industrial Engineering and Management, Technion-Israel Institute of Technology, Haifa, Israel.
- Cohen, A. and Mallows, C. (1980), "Analysis of Ranking Data," Technical memorandum, Bell Laboratories, Murray Hill, New Jersey.
- Critchlow, D. E. (1985), *Metric Methods for Analyzing Partially Ranked Data*, Lecture Notes in Statistics, 34, New York: Springer-Verlag.
- Critchlow, D. E. and Verducci, J. S. (1989), "Detecting a Trend in Paired Rankings," Technical Report No. 418, Dept. of Statistics, Ohio State Univ.
- Diaconis, P. (1988), *Group Representations in Probability and Statistics*, Hayward: Institute of Mathematical Statistics.
- McCullagh, P. (1990), "Models on Spheres and Models for Permutations," Technical Report, Department of Statistics, Univ. of Chicago.
- McCullagh, P. and Ye, J. (1990), "Matched Pairs and Ranked Data," Technical Report No. 287, Department of Statistics, Univ. of Chicago.
- Schulman, R. S. (1979), "A Geometric Model of Rank Correlation," *The Amer. Statist.*, 33 (2), 77-80.
- Thompson, G. L. (1992), "Generalized Permutation Polytopes and Exploratory Graphical Methods for Ranked Data," Unpublished manuscript.

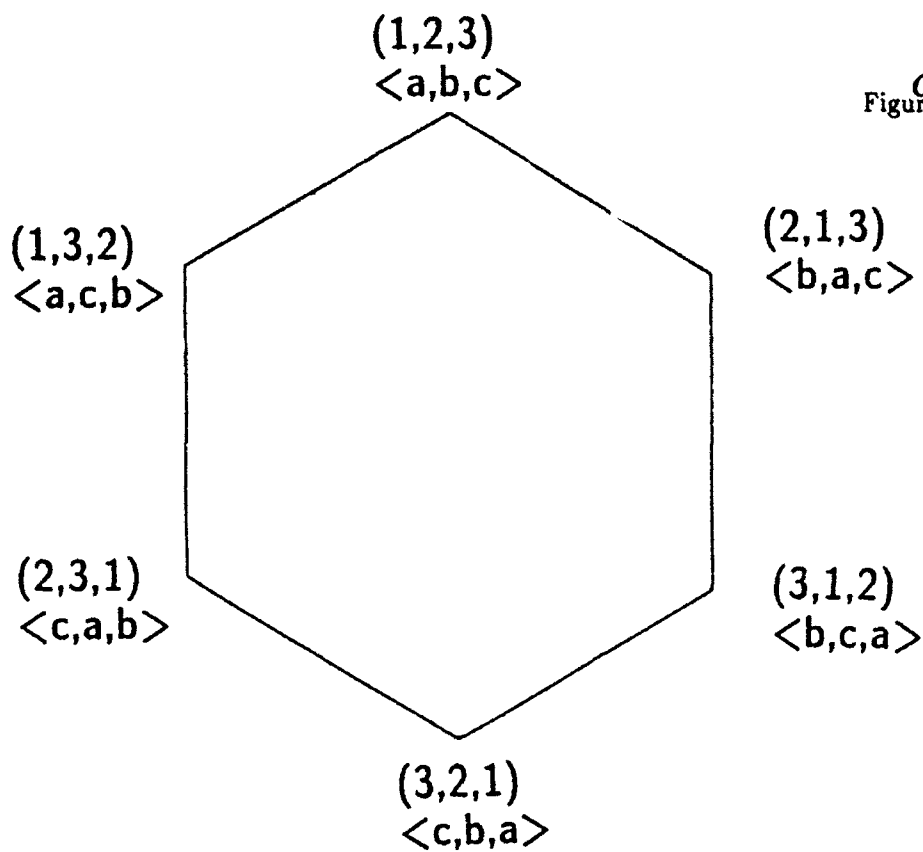
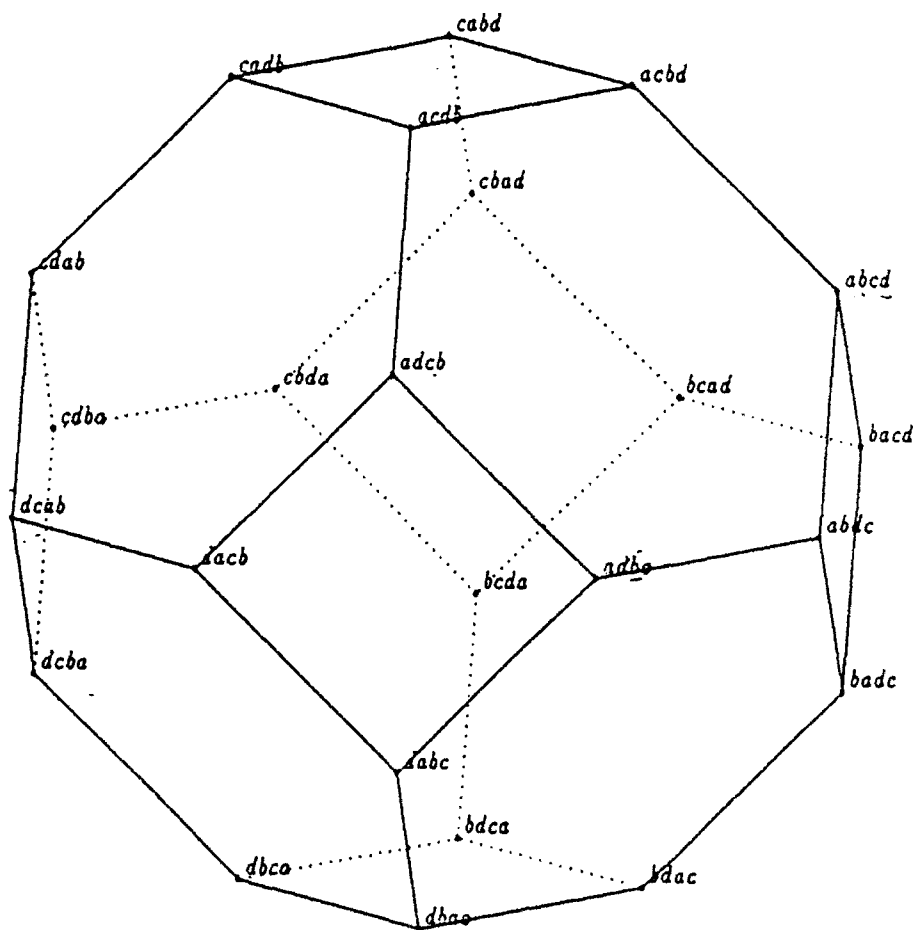


Figure 2
Truncated Octahedron



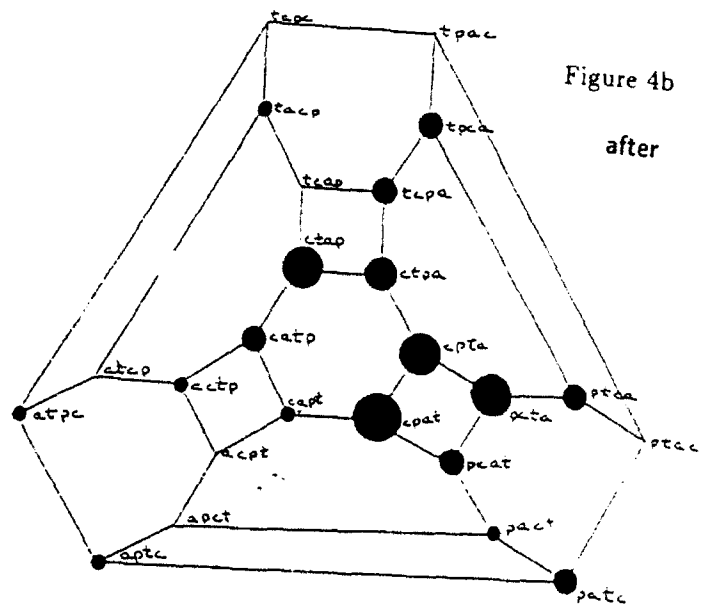
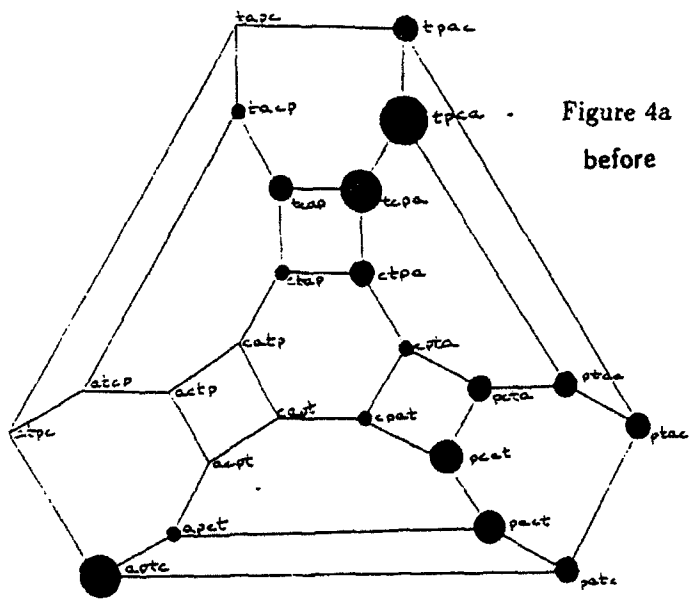
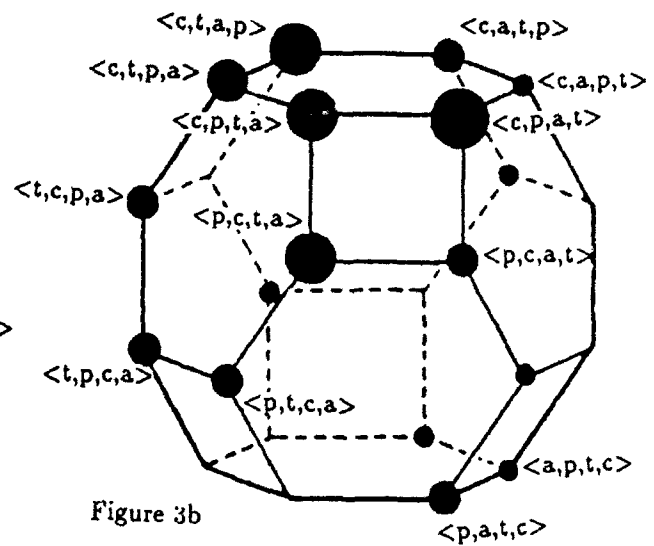
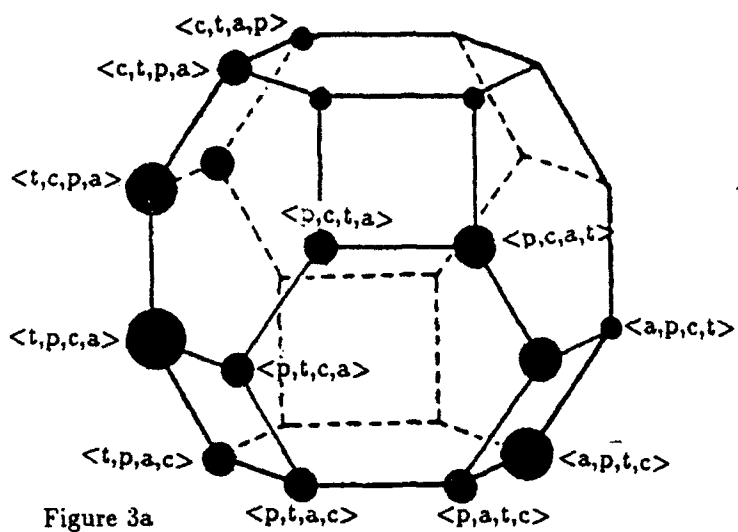


Figure 5a

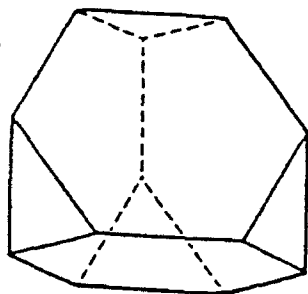


Figure 5c

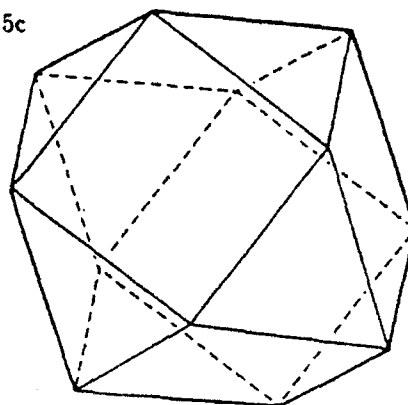


Figure 5b

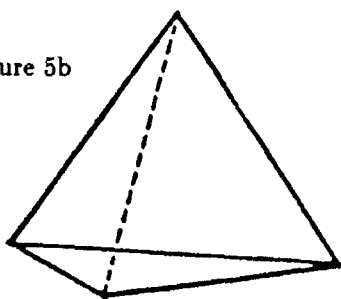
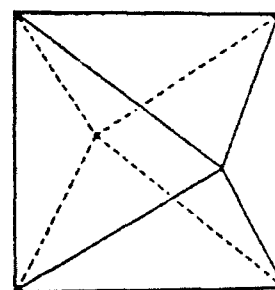


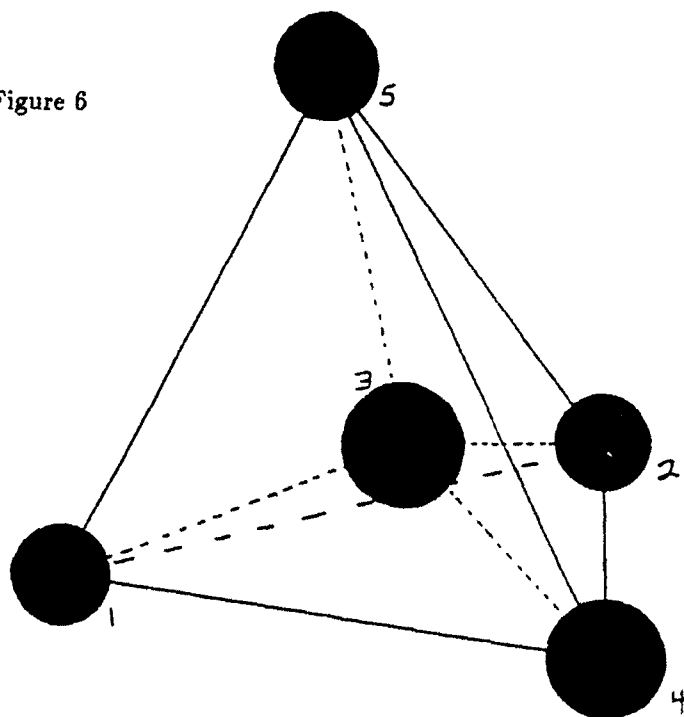
Figure 5d

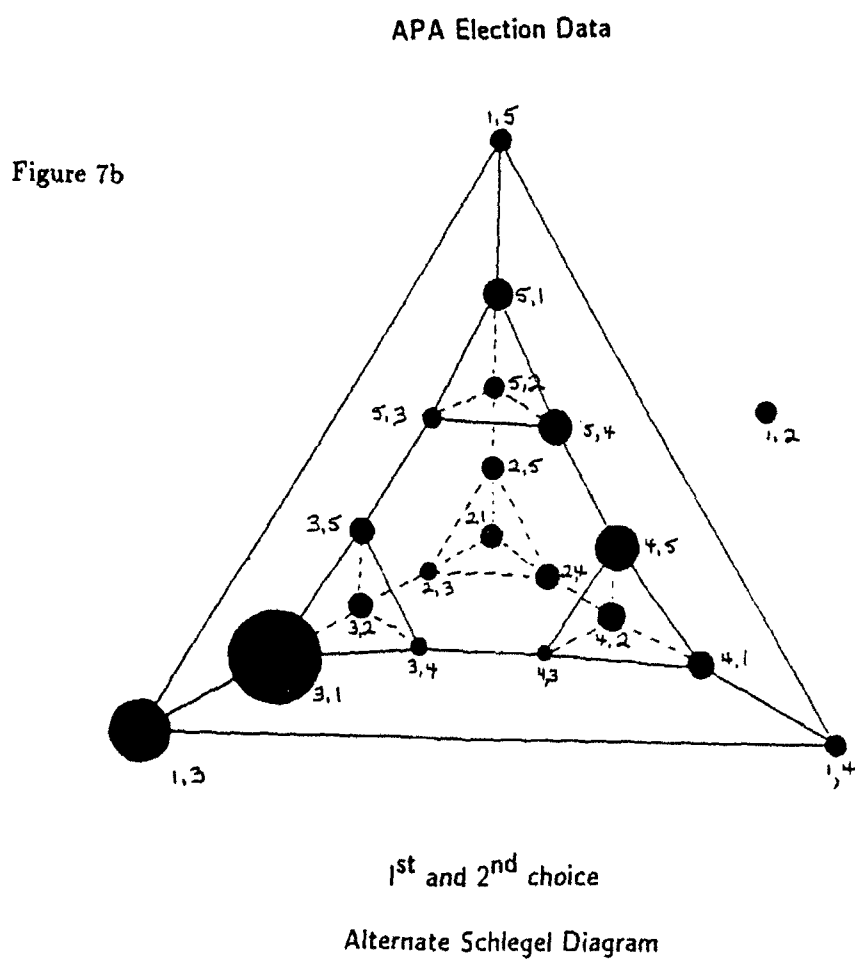
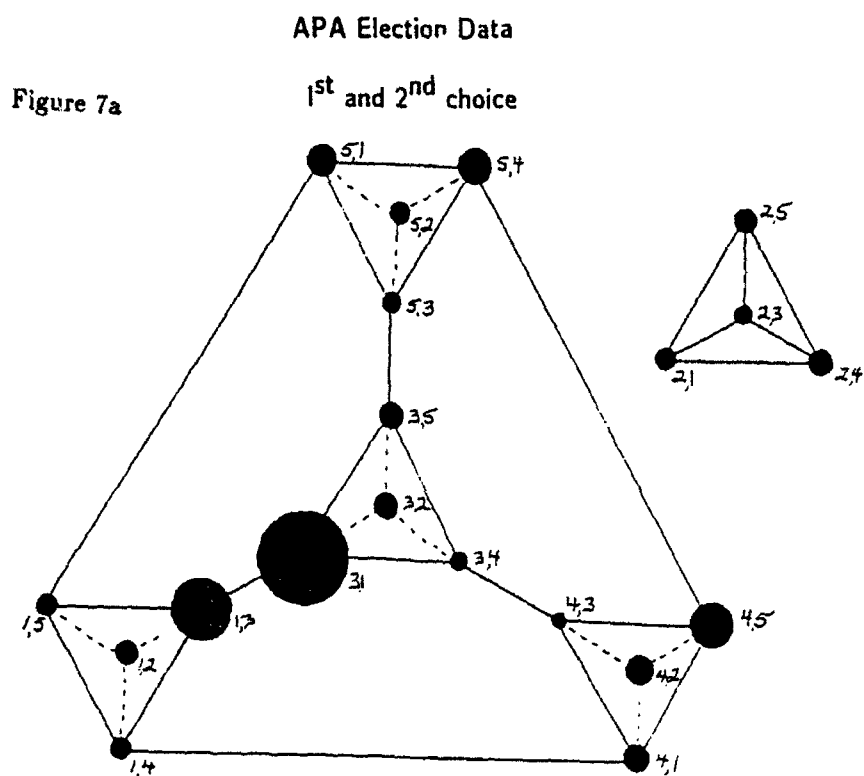


APA Election Data

1st choice

Figure 6





Methods for the Analysis of Coordinate Measurement Data

Fred L. Hulting
Alcoa Technical Center, D-AMCT
100 Technical Drive
Alcoa Center, PA 15069

Abstract

Measurement systems that record (x, y) or (x, y, z) coordinate data are widely used in industry to measure the dimensional characteristics of manufactured parts. Because successful monitoring and control of the manufacturing process depends, in part, on the validity of the measurements provided by these devices, periodic evaluations of measurement variation are required. These evaluations typically involve the statistical analysis of experimental data. While the coordinate measurements provided by these devices are inherently multidimensional, the data is often analyzed by applying traditional univariate statistical methods to each of the coordinate axes, or to one-dimensional summaries of the coordinate data. This reduction in dimensionality prior to analysis may produce misleading results, and it would be preferable to have methods which enable direct analysis of the original coordinate data. This paper considers one multivariate method obtained by extending traditional univariate methods based on random effects linear models. The method is illustrated using data from a study of the measurement variation in a coordinate measuring machine.

1. Introduction

In order to ensure the quality of manufactured parts, the location of features (e.g. holes, slots, surface points) on parts (e.g., subassemblies of automobiles or planes, electronic packages, etc.) are checked for conformance to specification during the manufacturing process. These checks involve recording the three-dimensional (x, y, z) coordinates of feature locations and comparing these measured coordinates to nominal (design) values. Generally, the measurements are taken by a computer-controlled device, such as a coordinate measuring machine or a vision system, and successful control of the manufacturing process depends, in part, on the validity of these measurements. Thus, periodic assessments of the performance of the measuring device are required. These assessments are typically based on the analysis of

data from experiments designed to address a number of measurement system characteristics, including accuracy (bias), precision (variation), calibration, resolution, and response time.

In assessing the precision of a measurement system, the sources that contribute to measurement variation are identified, and their contributions are quantified. Statistical methods for identifying and quantifying components of variation in one-dimensional data are well-developed, and are familiar to many practitioners (Hunter 1985). As a result, coordinate measurement data are often analyzed by applying these methods either to (i) each coordinate axis (x , y , and z), or to (ii) one-dimensional summaries of the coordinate data — e.g., the distance from the nominal value along a given direction, or the first principal component.

There are, however, potentially significant problems with these approaches. The first approach ignores relationships between the measurements along individual axes. As for the second approach, reducing the dimensionality of the multivariate data prior to analysis can result in lost information. The original coordinate data will generally be a richer source of information than either an individual axis or a one-dimensional summary, and one must question whether the variation in the summary data is fairly representing the variation in the measurement system. To avoid this problem, one can analyze the variation in the original coordinate data, perhaps using a multivariate method which is similar in spirit to one of the traditional methods for one-dimensional data. One such method has been proposed by Demeter (1989), although the method is limited in scope. A general statistical approach to the problem has not been developed.

This paper describes and illustrates one method for analyzing multivariate measurement data. The method is an extension of univariate analysis of variance methodology and is based on well-known results from linear models and multivariate statistics. The presentation is primarily expository, with technical details contained in the appendix. I will first introduce notation and discuss a model for the coordinate data. Then I discuss the de-

tails of the proposed analyses. Finally, I apply them to example data from a coordinate measuring machine.

These methods have other advantages besides protecting the information in the data. Often, part feature locations are toleranced using a geometric approach. This approach recognizes that deviations from nominal may occur in any direction, and it places engineering tolerance limits on the position of the feature, resulting in a spherical tolerance region. The multivariate analysis is a natural complement to geometric tolerancing because it enables the variation in all directions to be quantified simultaneously. This is useful when more than one axis or direction is to be considered, or when it is not known, *a priori*, which directions should be of concern.

The sources of variation that are to be studied will depend on the particular measurement system and its environment. Traditionally, in the manufacturing context, two sources of variation have been targeted for measurement systems: *within-operator* and *between-operator* (sometimes referred to as *repeatability* and *reproducibility*, respectively). The within-operator variation is that which is intrinsic to the use of the measurement system by a single operator (including "setup"). The between-operator variation is that which is due to differences in the way operators use the measurement system. Because this context is familiar to many practitioners, I will use it for this discussion. However, the methods I describe can be generalized to situations where other sources of variation are of interest.

2. Modeling Coordinate Data

In this section I will introduce the ideas behind collecting and modeling coordinate data for a measurement study. I will also exemplify this discussion with the description of a measurement study for a coordinate measuring machine (CMM).

2.1. Collecting Data

Data for a measurement study are often collected from a multi-factor experiment involving one or more random factors. In the context considered here, there are usually two factors: parts and operators. Consider an experiment in which I parts are measured K times by each of J operators. Typically, a "run" in this experiment would consist of an operator loading a part into the measurement device and then recording measurements for some number of features (L) on the part. Let $u_{ijkl} = (x_{ijkl}, y_{ijkl}, z_{ijkl})$ be the coordinate of the l^{th} feature recorded on the k^{th} measurement of the i^{th} part by the j^{th} operator. This observed value can be written as

$$u_{ijkl} = n_l + d_{ijkl},$$

where $n_l = (x_l^0, y_l^0, z_l^0)'$ is the nominal (design) coordinate for the l^{th} feature, and

$$d_{ijkl} = u_{ijkl} - n_l$$

is the observed deviation from nominal. Subsequently, we will treat the d_{ijkl} as the observed data. Furthermore, unless noted, we will assume that we are focusing on data for a single feature, and drop the subscript l .

Generally, both parts and operators are treated as random factors: the parts are assumed to be drawn randomly from the process that produces them, and the operators represent a sample from a larger pool of workers who operate the measurement system. In some instances there may, in fact, be very few operators, in which case the operators are treated as a fixed factor. In this paper, operators will be treated as a random factor. Modifications to accommodate the fixed factor are straightforward.

CMM Example: There are many types of measurement devices that yield two- or three-dimensional coordinate data. This paper was motivated by work with CMMs. A CMM is a computer-controlled device that uses a contact probe mounted on the end of a robotic arm to record data. The CMM tracks the location (x, y, z) of the probe tip as it moves in three-dimensional space. Measurements are made by noting the coordinates of the probe when it makes contact with the part feature of interest. In the automobile industry, CMMs are used to measure the locations of various features on complete bodies, subassemblies, production parts, fixtures, dies, etc.

Consider data from a CMM measurement study involving a sheet metal panel for an automobile. Twenty features on ten parts were each measured five times by three operators. So, in our notation, $I = 10$, $J = 3$, $K = 5$, and $L = 20$. Only one of the features — a hole center — is discussed here. The data were coded so that the nominal coordinate is $(0, 0, 0)$. The units of measurement are millimeters.

2.2. Modeling One-dimensional Data

Before considering a model for the observed deviations d_{ijk} , recall one univariate approach to modeling data from a measurement study. Let w_{ijk} be a one-dimensional measurement, and assume that the w_{ijk} are random quantities that follow the random effects linear model

$$w_{ijk} = \mu + \alpha_i + \beta_j + \lambda_{ij} + \gamma_{ijk}, \quad (1)$$

which can be thought of as

$$\begin{aligned} w_{ijk} = & \text{(part mean + measurement bias) +} \\ & \text{part effect + operator effect +} \\ & \text{+ part-operator interaction effect + error.} \end{aligned}$$

The unknown parameter μ represents the sum of the true mean for the part and the bias of the measurement system. The independent random effects α_i , β_j , and λ_{ij} represent the deviations from μ that are due to parts, operators, and the interaction of parts and operators, respectively. The γ_{ijk} are considered independent random "errors" associated with the repeats by a single operator. In this model, the measurement errors are represented by $e_{ijk} = \beta_j + \lambda_{ij} + \gamma_{ijk}$. We assume that $\alpha_i \sim N(0, \sigma_p^2)$, $\beta_j \sim N(0, \sigma_{oo}^2)$, $\lambda_{ij} \sim N(0, \sigma_{po}^2)$, and $\gamma_{ijk} \sim N(0, \sigma_r^2)$, where " $N(\cdot, \cdot)$ " refers to a normal (Gaussian) probability distribution. The γ_{ijk} are assumed to be independent of the other random effects. Thus, $e_{ijk} \sim N(0, \sigma_{oo}^2 + \sigma_{po}^2 + \sigma_r^2)$. The variances σ_p^2 , σ_{oo}^2 , and σ_{po}^2 are assumed to be ≥ 0 ; σ_r^2 is assumed to be > 0 .

The goal of our analysis is to characterize the components that contribute to the variation in our observed measurements. Under this model, those components are represented by functions of σ_r^2 , σ_p^2 , σ_{oo}^2 , and σ_{po}^2 . Specifically, we are interested in: $\sigma_o^2 = \sigma_{oo}^2 + \sigma_{po}^2$, the between-operator variance; σ_r^2 , the variance in repeat measurements by a single operator; and $\sigma_e^2 = \sigma_o^2 + \sigma_r^2$, the total measurement error variation. Assessments of measurement variation will be based on estimates of these variances obtained from the experimental data.

The univariate model is applied to coordinate data by taking w_{ijk} to be a real-valued linear function of the elements of d_{ijk} . That is, $w_{ijk} = \eta' d_{ijk}$ where η is a known 3×1 vector. For example, $w_{ijk} = x_{ijk} - x^0$ is the deviation from nominal along x -axis. Another commonly used summary is the *distance from nominal* $w_{ijk} = v' d_{ijk}$, where v is a unit-length directional vector. This is just the projection of the deviation from nominal d_{ijk} onto an axis defining a particular direction in coordinate space. Typically, the direction is chosen based on engineering concerns; for example, the vector may be normal to the surface containing the feature of interest. However, one might also choose the vector v based on a principal components analysis of the data (see, for example, Jackson 1981). The usual analysis of the CMM hole center data is based on this latter type of transformation. However, in the case of the CMM hole center data, the distance from nominal is simply the euclidean distance $(d'_{ijk} d_{ijk})^{1/2}$.

2.3 Modeling Multidimensional Data

One approach to modeling multidimensional data is to use a multivariate generalization of the univariate model (1). Let w_{ijk} be any real- or vector-valued linear function of the coordinate data, that is, $w_{ijk} = Y d_{ijk}$, where Y is a known $m \times 3$ matrix ($1 \leq m \leq 3$). For example, both the original d_{ijk} ($Y = I$), and the distances from nominal $v' d_{ijk}$ ($Y = v'$), are of this form. Assume that the w_{ijk} follow the *two-way (with interaction) m-*

dimensional random effects linear model:

$$w_{ijk} = \mu + \alpha_i + \beta_j + \lambda_{ij} + \gamma_{ijk} \quad (2)$$

Here μ is an unknown $m \times 1$ parameter vector, and α_i , β_j , λ_{ij} , and γ_{ijk} are independent $m \times 1$ random vectors such that $\alpha_i \sim N_m(0, \Sigma_p)$, $\beta_j \sim N_m(0, \Sigma_{oo})$, $\lambda_{ij} \sim N_m(0, \Sigma_{po})$, and $\gamma_{ijk} \sim N_m(0, \Sigma_r)$. The notation " N_m " refers to an m -dimensional normal distribution (e.g., Johnson and Wichern, p. 121). This model is the natural generalization of model (1), and the random effects and parameters are interpreted in the same manner as their univariate counterparts. In particular, the random vector $e_{ijk} = \beta_j + \lambda_{ij} + \gamma_{ijk}$ represents the measurement error, with $e_{ijk} \sim N_m(0, \Sigma_{oo} + \Sigma_{po} + \Sigma_r)$.

The primary difference between the two models is that model (2) uses covariance matrices to describe variation in several dimensions while model (1) uses variances to describe variation in a single dimension. These matrices are more difficult to interpret than simple variances, but the focus of our analysis does not change. The quantities of interest are still: the between-operator variation (characterized by $\Sigma_o = \Sigma_{oo} + \Sigma_{po}$), the within-operator variation (characterized by Σ_r), and the total measurement error variation (characterized by $\Sigma_e = \Sigma_o + \Sigma_r$). However, because the covariance matrices are difficult to interpret, it will be necessary to develop useful numerical and graphical summaries of these matrices.

CMM Example: Initial analyses showed the variation along the z -axis to be negligible compared to the variation along the x and y axes. This was expected because the z axis is aligned with the surface containing the hole prior to measurement. Given this knowledge, it is reasonable to use only the x and y data in the analysis. That is, we will treat the

$$w_{ijk} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \end{pmatrix} d_{ijk} = \begin{pmatrix} x_{ijk} \\ y_{ijk} \end{pmatrix}$$

as the data to be analyzed under model (2).

3. The Analysis of Coordinate Data

As with model specification, the methods I propose for analyzing multivariate measurement variation are natural generalizations of commonly used univariate procedures. To motivate the multivariate methods, this section begins with a brief description of an approach to analyzing variation in one dimension.

3.1. One-Dimensional Approach

For the one-dimensional problem, the variance components completely characterize the variation of the measurement system (assuming the model is correct). Of course, these variances are unknown. In practice, we use

estimates of the variance components — denoted $\hat{\sigma}_p^2$, $\hat{\sigma}_o^2$, $\hat{\sigma}_r^2$, and $\hat{\sigma}_e^2$ — in place of the true quantities and proceed as if σ_p^2 , σ_o^2 , σ_r^2 , and σ_e^2 are known. For balanced data, the variance estimates can be easily obtained using the analysis-of-variance (ANOVA) method (Hunter 1985; Montgomery 1991, pp. 222–224).

Once we have estimates of the variance components, the next step is to determine whether the measurement variation is sufficiently small. Traditionally, this determination is made by comparing the “size” of the measurement variation to the size of the tolerance region for the part to be measured. This is one way to quantify the ability of the measurement system to correctly identify parts that are out of specification. Let k_α be the value such that $\text{Prob}\{-k_\alpha/2 \leq Z \leq k_\alpha/2\} = 1 - \alpha$, where Z is a $N(0, 1)$ random variable. For example, $k_{0.01} = 5.15$. Also, let TOL be the width of the engineering-determined tolerance interval. A comparison of measurement variation to tolerance is given by the *variance-to-tolerance ratio* $R_e(\alpha) = k_\alpha \hat{\sigma}_e / \text{TOL}$. Note that the middle $100(1 - \alpha)\%$ of the normal distribution with variance $\hat{\sigma}_e^2$ is an interval of width $k_\alpha \hat{\sigma}_e$. So, $R_e(\alpha)$ makes the natural comparison of the lengths of the two intervals and may be interpreted as the amount of the tolerance region that is occupied by the “bulk” of the measurement error distribution. Comparable measures involving $\hat{\sigma}_o$ and $\hat{\sigma}_r$ [i.e., $R_o(\alpha)$ and $R_r(\alpha)$] can be defined.

Often, suitability of a measurement system is determined by comparing $R_e(\alpha)$, or a similar measure, to a standard or corporate “rule-of-thumb,” derived from past experience. For example, $R_e(\alpha) \leq 0.10$ might imply that the measurement system is satisfactory. Such comparisons are essentially probability statements. Letting ϵ represent the true, but unobservable, measurement error, the statement “ $R_e(\alpha) < 0.10$ ” is equivalent to the statement “ $\text{Prob}[|\epsilon| \geq (0.10 \times \text{TOL})] \leq \alpha$.”

Comparisons of measurement variation to part tolerances are not very useful to an organization practicing continuous process improvement. The reason for this is that, from a process monitoring or process control point of view, it is more important to have the measurement variation — represented by $\hat{\sigma}_e$, $\hat{\sigma}_o$, and $\hat{\sigma}_r$ — be small relative to the actual process variation — represented by $\hat{\sigma}_p$. Thus, we would like to have measures which compare measurement variation and process variation. The signal-to-noise (SN) type ratios $\hat{\sigma}_p/\hat{\sigma}_e$, $\hat{\sigma}_p/\hat{\sigma}_o$, and $\hat{\sigma}_p/\hat{\sigma}_r$ are one choice. Desirable values of the SN ratios are those that are much greater than one. Another commonly used measure is the efficiency ratio $\text{EFF} = \hat{\sigma}_p^2 / (\hat{\sigma}_p^2 + \hat{\sigma}_e^2)$, which compares the actual process variation with the observed process variation.

3.2. A Multi-dimensional Approach

The simplest way to extend from one-dimension to several dimensions is to simultaneously consider many one-dimensional summaries of the coordinate data. Because each summary of the form $w_{ijk} = \eta' d_{ijk}$ is a projection onto a vector in the three-dimensional coordinate space, analyzing different summaries provides one method of examining the variation along different directions in the coordinate space. This “one-at-a-time” multidimensional approach may be appealing to many practitioners because it relies on familiar quantities such as the variance components and associated ratios. However, it is not a general framework for multidimensional data.

To move to a fully multidimensional approach, we must abandon the easily-interpreted variances and work to derive summaries of the covariance matrices. I will derive extensions of $R_e(\alpha)$ and EFF under the assumption that model (2) is adequate, and that the covariance matrices have been estimated. As before, I will proceed as if the covariance matrices are known. Procedures for estimating the covariance matrices can be adapted from Amemiya (1985) and Calvin and Dykstra (1991). One procedure is described in Appendix A. Computational expressions for the quantities discussed here may be found in Appendix B.

The key to the extension of $R_e(\alpha)$ is the one-to-one correspondence between a covariance matrix Σ and the $100(1 - \alpha)\%$ elliptical contour of the $N_m(0, \Sigma)$ distribution. Represent the measurement error covariance matrix Σ_e by the corresponding $100(1 - \alpha)\%$ contour, and assume a spherical tolerance region. Then, a size comparison of the measurement variation and the tolerance can be thought of as a size comparison between an ellipsoid and a sphere. Two ratios that compare the sizes of ellipsoids and spheres are:

$$R_{e(V)}(\alpha) = \frac{\text{Vol of } 100(1 - \alpha)\% N_m(0, \Sigma_e) \text{ ellipsoid}}{\text{Vol of tolerance sphere}}$$

and

$$R_{e(L)}(\alpha) = \frac{\text{Max length of } 100(1 - \alpha)\% N_m(0, \Sigma_e) \text{ ellipsoid}}{\text{Diameter of tolerance sphere}}$$

The first ratio is a comparison based on volume (V); the other is based on length (L).

In the preceding discussion, we have assumed that the tolerance regions are spherical, as might be derived from a geometric tolerancing approach. If the shape of the tolerance region is not spherical, appropriate modifications should be made.

Now consider the extension of efficiency ratio EFF to more than one dimension. EFF can be viewed as a comparison of the lengths of the two intervals that represent

the middle $100(1-\alpha)\%$ of the $N(0, \hat{\sigma}_p^2)$ and $N(0, \hat{\sigma}_p^2 + \hat{\sigma}_e^2)$ distributions, respectively. The multivariate counterparts to these two intervals are the two ellipsoids that represent the middle $100(1-\alpha)\%$ of the $MVN(0, \hat{\Sigma}_p)$ and $MVN(0, \hat{\Sigma}_p + \hat{\Sigma}_e)$ distributions, respectively. Extensions of EFF may be based on a ratio that compares the sizes of these ellipsoids. As with $R_{e(V)}(\alpha)$ and $R_{e(L)}(\alpha)$, the comparison of size can be based on volume or length. An extension based on a volume comparison is

$$EFF(V) = \left(\frac{\text{Vol of } 100(1-\alpha)\% \text{ } N_m(0, \hat{\Sigma}_p) \text{ ellipsoid}}{\text{Vol of } 100(1-\alpha)\% \text{ } N_m(0, \hat{\Sigma}_p + \hat{\Sigma}_e) \text{ ellipsoid}} \right)^2.$$

An extension based on a length comparison is not as straightforward. This is because the ellipses may not be oriented in the same direction and a comparison of the main axes would not be meaningful. In situations such as this, "length" comparisons can be made along specific directions. Letting g be a (unit length) vector that defines a chosen direction, we have

$$EFF(L)(g) = \left(\frac{\text{Length of } 100(1-\alpha)\% \text{ } N_m(0, \hat{\Sigma}_p) \text{ ellipsoid}}{\text{Length of } 100(1-\alpha)\% \text{ } N_m(0, \hat{\Sigma}_p + \hat{\Sigma}_e) \text{ ellipsoid}} \right)^2,$$

where the lengths are taken along g . Two interesting choices for g are those vectors which yield the maximum and minimum values of $EFF(L)(g)$. Note that although these definitions depend on α , the values of $EFF(V)$ and $EFF(L)(g)$ are invariant to the choice of α .

3.3. Graphical Methodology

It should be clear that the numerical summaries of the covariance matrices will generally be insufficient for a thorough analysis of measurement variation. Contour plots can form the basis for useful data displays that supplement those summary statistics. A particular contour, say the 99% contour, is enhanced by plotting "data points" that have a sample covariance matrix equal to the estimated covariance matrix which defines the contour. Such displays are referred to as "contour + data" plots.

The 99% contour associated with $\hat{\Sigma}_p$ would be accompanied by the residuals obtained from fitting model (2). The 99% contour computed from $\hat{\Sigma}_e$ would be enhanced by "measurement error residuals," which are defined in Appendix C. It is not possible to define suitable data points to accompany the contours for $\hat{\Sigma}_p$ and $\hat{\Sigma}_e$. In that case one could simply overlay the means for the parts or operators, respectively. When the dimension m is 2, each contour+data plot is constructed using a single two-dimensional scatter plot. When $m = 3$, the more appropriate display consists of the three pairwise scatterplots x - y , z - y , and x - z , arranged in a scatterplot matrix.

4. Example: Coordinate Measuring Machines

We now turn to an analysis of the hole center data from the CMM experiment described in Section 2. The contour+data plots are given in Figure 1. Tables 1 and 2 summarize the statistics for the two-dimensional analysis. Figures 2 and 3 offer visual interpretations of the ratios defined in Section 3. The positional tolerance region used in these plots is a circle of radius 0.3 mm. In Figures 2 and 3 a line has been drawn to indicate the direction used for length comparisons.

In Figure 1c, the principal axes of the within-operator ellipse are nearly aligned with the x and y axes, respectively, suggesting independence of the x and y within-operator measurement errors. In Figures 1d and 2, however, the total measurement variation ellipse is oriented along an angle of about 20 degrees. This change in orientation is due to the difference between operators that is clearly depicted in Figure 1b. It is also notable that the principal axis of the part ellipse is oriented differently than the measurement error ellipses, as seen in Figure 3. This would not have been revealed had the data been reduced to a single dimension prior to analysis.

What can be said about the variation in the measurement system? Certainly some improvements are needed. The efficiency volume ratio is $EFF(V) = 0.24$, while the efficiency length ratio can be as small as $EFF(L)(g) = 0.37$ (for $g = (-0.99, -0.14)'$, the main axis of the $\hat{\Sigma}_p + \hat{\Sigma}_e$ ellipse) — see Figure 3. These values suggest that the part variation is overshadowed by the measurement variation, and thus the measurement system is incapable of monitoring the process. Because only three operators were involved in the study, one should be careful about generalizing these results — the between-operator "variation" may simply reflect systematic differences between these three particular operators. However, the fact that the largest component of the measurement variation is due to between-operator differences suggests that operator training and/or a fixture redesign may improve the results. Indeed, it was later determined that the observed operator variation could largely be attributed to differences in the manner in which the three operators mounted and fixtured the panel.

5. Summary

Coordinate measurement systems are commonly used in industry, and there has been little work concerning the analysis of data arising from these devices. In this paper a multivariate extension of the traditional univariate method for assessing measurement variation is proposed, and it is illustrated by an application to coordinate measuring machines. Simple numerical and graphical sum-

maries are developed for conveying the results of the analysis. The method provides a more complete characterization of measurement system performance than the traditional univariate approach.

Acknowledgements

This work was initiated while the author was a member of the Mathematics Department of General Motors Research Laboratories. Helpful discussions with Jeff Robinson and Laslow Demeter are gratefully acknowledged. The methods in this paper have been implemented in the S language (Becker, Chambers, and Wilks 1988), and the software is available from the author on request.

REFERENCES

- Alt, F. B. (1985). "Multivariate Quality Control." In *Encyclopedia of Statistical Sciences*, eds. S. Kotz and N. L. Johnson. Wiley, New York, Vol. 6, pp. 111-122.
- Amemiya, Y. (1985). "What Should Be Done When An Estimated Between-Group Covariance Matrix Is Not Nonnegative Definite?" *The American Statistician* 39, pp. 112-118.
- Anderson, T. W. (1985). "Components of Variance in MANOVA." In *Multivariate Analysis - VI*, P. R. Krishnaiah, ed. Elsevier, New York.
- Becker, R. A., Chambers, J. M., and Wilks, A. R. (1988). *The New S Language*. Wadsworth & Brooks/Cole, Pacific Grove, CA.
- Calvin, J. A., and Dykstra, R. L. (1991). "Least Squares Estimation of Covariance Matrices in Balanced Multivariate Variance Components Models." *Journal of the American Statistical Association*, 86, 388-395.
- Demeter, L. (1989). "Gauge Repeatability and Reproducibility: Improvements Needed to Elevate the State of the Art to Meet Evolving Industry Standards." Technical Paper 890772. *Society of Automotive Engineers*.
- Hunter, J. S. (1985). "Measurement Error." In *Encyclopedia of Statistical Sciences*, eds. S. Kotz and N. L. Johnson. Wiley, New York, Vol 5, pp. 378-381.
- Jackson, J. E. (1956). "Quality Control Methods for Several Related Variables." *Technometrics* 7, pp. 4-8.
- Jackson, J. E. (1981). "Principal Components and Factor Analysis: Part I — Principal Components." *Journal of Quality Technology* 12, pp. 201-213.
- Johnson, R. A., and Wichern, D. W. (1988). *Applied Multivariate Statistical Analysis*. Prentice-Hall, Englewood Cliffs, NJ.
- Montgomery, D. C. (1991). *Design and Analysis of Experiments*, 3rd edition. Wiley, New York.

Appendix A. Estimation of the Covariance Matrices

In this Appendix I describe one method for estimating the covariance matrix components of model (2). The total (corrected) variability in the data is $S_T = \sum_{i=1}^I \sum_{j=1}^J \sum_{k=1}^K (w_{ijk} - \bar{w}_{i..})(w_{ijk} - \bar{w}_{i..})'$, where the dot notation is used to denote sums over subscripts, and the bars denote means. This variability can be partitioned as $S_T = S_p + S_{oo} + S_{po} + S_r$ where $S_p = JK \sum_{i=1}^I (\bar{w}_{i..} - \bar{w}_{...})(\bar{w}_{i..} - \bar{w}_{...})'$, $S_{oo} = IK \sum_{j=1}^J (\bar{w}_{.j.} - \bar{w}_{...})(\bar{w}_{.j.} - \bar{w}_{...})'$, $S_{po} = K \sum_{i=1}^I \sum_{j=1}^J (\bar{w}_{ij.} - \bar{w}_{i..} - \bar{w}_{.j.} + \bar{w}_{...})(\bar{w}_{ij.} - \bar{w}_{i..} - \bar{w}_{.j.} + \bar{w}_{...})'$, and $S_r = \sum_{i=1}^I \sum_{j=1}^J \sum_{k=1}^K (w_{ijk} - \bar{w}_{ij.})(w_{ijk} - \bar{w}_{ij.})'$. Such a partitioning of the variability in a data set can be summarized in an ANOVA table. The ANOVA table for this partitioning is shown in Table 3. The expected mean squares are computed under the assumptions associated with model (2).

The ANOVA table suggests the usual method-of-moments procedure for estimating the covariance matrices Σ_p , Σ_{oo} , Σ_{po} , and Σ_r — equate the mean square matrices to their expectations and solve for the covariance matrices. The estimators obtained in this way are $\hat{\Sigma}_p = (1/JK)[M_p - M_{po}]$, $\hat{\Sigma}_{oo} = (1/IK)[M_{oo} - M_{po}]$, $\hat{\Sigma}_{po} = (1/K)[M_{po} - M_r]$, and $\hat{\Sigma}_r = M_r$. The same method can be used to obtain estimators of sums of these covariance matrices, e.g., $\hat{\Sigma}_o = (1/IK)[M_{oo} + (I-1)M_{po} - IM_r]$, and $\hat{\Sigma}_e = (1/IK)[M_{oo} + (I-1)M_{po} + I(K-1)M_r]$. Note that all estimators are linear combinations of the mean square matrices from our ANOVA. Also, as with our model, this estimation procedure reduces, in the case of $p = 1$, to the usual estimation procedures for the univariate case.

Although not stated in the text, it is assumed that Σ_r is positive definite (PD) and Σ_p , Σ_{oo} , and Σ_{po} are nonnegative definite (NND). However, it is possible for the estimates $\hat{\Sigma}_p$ and $\hat{\Sigma}_o$ to be negative definite. In that case, the estimates must be adjusted so that they lie in the parameter space. One way to do this is to extend the procedure suggested by Anderson (1984) and Amemiya (1985) for the one-way multivariate random effects linear model. Let M_A be a $m \times m$ NND matrix and M_B be a $m \times m$ PD matrix. Assume that a method-of-moments estimator of a covari-

ance matrix Σ is of the form $\hat{\Sigma} = C^* [M_A - M_B]$ for some constant C^* , and that the estimate obtained for the current data set is not NND. Using the characteristic roots and vectors associated with the determinantal equation $|M_A - \eta M_B| = 0$, it is possible to write $M_A - M_B = [M_A - M_b]^+ + [M_A - M_b]^-$, where $[M_A - M_b]^+$ is a NND matrix, and $[M_A - M_b]^-$ is a negative definite matrix. The matrix $[M_A - M_b]^+$ is "closest" to $[M_A - M_B]$ among all NND matrices having the same characteristic vectors in a certain metric. Since $[M_A - M_b]^+ = M_A - M_B$ when $b = m$, it appears natural to use the following as the estimator of Σ :

$$\hat{\Sigma}^+ = \begin{cases} C^* [M_A - M_B]^+ & 0 < b \leq m \\ 0 & b = 0 \end{cases}$$

The details of the procedure may be found in Amemiya (1985).

To apply this general procedure, it is only necessary to identify, for each linear combination of mean square matrices, the constant C^* and the matrices M_A and M_B . However, the matrix playing the role of M_B may not be PD for some of the linear combinations considered here. To correct this, I substitute a PD estimate of $E(M_B)$ for M_B , as suggested by Amemiya (1985).

Appendix B. Computational Expressions for

$R_e(V)(\alpha)$, $R_e(L)(\alpha)$, $\text{EFF}(V)$, and $\text{EFF}(L)(g)$

Let $\delta_1^{(e)} \geq \dots \geq \delta_m^{(e)}$ be the eigenvalues of $\hat{\Sigma}_e$ and let $l_1^{(e)}, \dots, l_m^{(e)}$ be the associated eigenvectors. The "volume" of the $100(1 - \alpha)\%$ ellipsoid for a $N_m(0, \hat{\Sigma}_e)$ distribution is given by $V^{(e)}(\alpha) = \left(\prod_{i=1}^m \delta_i^{(e)}\right)^{1/2} (2[\pi\chi_m^2(\alpha)]^{m/2}/[m\Gamma(m/2)])$, where $\chi_m^2(\alpha)$ is the upper- α point of the chi-squared distribution with m degrees of freedom, and $\Gamma(\cdot)$ is the gamma function (Johnson and Wichern 1988, p. 103). Also, the lengths of the principal axes of that ellipsoid (which are defined by the $l_i^{(e)}$) are $\nu_i^{(e)}(\alpha) = [\delta_i^{(e)}]^{1/2} \times 2[\chi_m^2(\alpha)]^{1/2}$ (Johnson and Wichern 1988, p. 126).

Now, assume a spherical tolerance region with diameter TOL. Then, the two extensions of $R_e(\alpha)$ are $R_e(V)(\alpha) = V^{(e)}(\alpha)/\text{VOL}$, where VOL is TOL (for $m = 1$), $\pi(\text{TOL})^2/4$ (for $m = 2$), or $\pi(\text{TOL})^3/6$ (for $m = 3$), and $R_e(L)(\alpha) = \nu_1^{(e)}(\alpha)/\text{TOL}$.

Now, let $\delta_1^{(p)} \geq \dots \geq \delta_m^{(p)}$ be the eigenvalues of $\hat{\Sigma}_p$ and let $\delta_1^{(p+e)} \geq \dots \geq \delta_m^{(p+e)}$ be the eigenvalues of $\hat{\Sigma}_p + \hat{\Sigma}_e$. Then, $\text{EFF}(V) = \left(\prod_{i=1}^m \delta_i^{(p)} / \prod_{i=1}^m \delta_i^{(p+e)}\right)^2$. The computation of $\text{EFF}(L)(g)$ is more complicated. First, polar

coordinate representations of the two 99% ellipse associated with $\hat{\Sigma}_p$ and $\hat{\Sigma}_p + \hat{\Sigma}_e$ are computed. Denote these as $\{\theta, r^{(p)}(\theta); 0 \leq \theta \leq 2\pi\}$ and $\{\theta, r^{(p+e)}(\theta); 0 \leq \theta \leq 2\pi\}$, respectively. Letting $\theta(g)$ be the angle corresponding to a directional vector g , it follows that $\text{EFF}(L)(g) = (\{r^{(p)}[\theta(g)]\} / \{r^{(p+e)}[\theta(g)]\})^2$.

Appendix C. Measurement Error Residuals

Recall, from Appendix A, that $\hat{\Sigma}_e = (1/IK)[M_{oo} + (I - 1)M_{po} + I(K - 1)M_r]$. Letting $s_{ijk} = (\bar{w}_{ij} - \bar{w}_{i..}) + [(J - 1)/J]^{1/2}(w_{ijk} - \bar{w}_{ij.})$, $\hat{\Sigma}_e$ can be rewritten as $[IJ(K - 1)]^{-1} \sum_{i=1}^I \sum_{j=1}^J \sum_{k=1}^K s_{ijk} s'_{ijk}$. So, treating the s_{ijk} as data, the sample average would be 0 and the sample covariance matrix would be $\{[IK(J - 1)]/[IJK - 1]\} \hat{\Sigma}_e$. Thus, a reasonable choice for data points to accompany the contour plots derived from $\hat{\Sigma}_e$ is $\hat{e}_{ijk} = \{[IK(J - 1)]/[IJK - 1]\} s_{ijk}$. Note that \hat{e}_{ijk} represents the major contribution of w_{ijk} to the estimate $\hat{\Sigma}_e$. Thus, the \hat{e}_{ijk} might be used to identify influential w_{ijk} values.

Table 1. Results from Two-dimensional (x, y) Analysis of Hole Center Data

Covariance Matrices		Axes of Ellipse		Axis Length
$\hat{\Sigma}_p \times 10^4$		$l_1^{(p)}$	$l_2^{(p)}$	$\nu^{(p)}$
9.76	-3.88	-0.879	-0.476	0.209
-3.88	4.70	0.476	-0.879	0.098
$\hat{\Sigma}_o \times 10^3$		$l_1^{(o)}$	$l_2^{(o)}$	$\nu^{(o)}$
1.85	0.80	0.887	-0.461	0.289
0.80	0.73	0.461	0.887	0.108
$\hat{\Sigma}_r \times 10^4$		$l_1^{(r)}$	$l_2^{(r)}$	$\nu^{(r)}$
8.91	-0.78	-0.994	-0.113	0.182
-0.78	2.19	0.113	-0.994	0.088
$\hat{\Sigma}_e \times 10^3$		$l_1^{(e)}$	$l_2^{(e)}$	$\nu^{(e)}$
2.74	0.72	-0.944	0.331	0.332
0.72	0.90	-0.331	-0.944	0.160

Table 2. Ratio Estimates from Analysis of Hole Center Data ($\alpha = 0.01$)

Source of Variation	Comparisons to Tolerance Region		Comparisons to Total Variation	
	Volume	Length	Volume	Length ⁽¹⁾
Part	$R_p(V) = 0.06$	$R_p(L) = 0.35$	$EFF_{(V)} = 0.24$	$EFF_{(L)}(g) = 0.37$
Between Operator	$R_o(V) = 0.09$	$R_o(L) = 0.48$	0.37	0.60
Within Operator	$R_r(V) = 0.04$	$R_r(L) = 0.30$	0.19	0.44
Within + Between	$R_e(V) = 0.15$	$R_e(L) = 0.55$	0.63	0.84

(1) Comparison along main axis of $\Sigma_e + \Sigma_p$ ellipse, i.e., $g = (-0.99, -0.14)'$

Table 3. An ANOVA for the Multivariate Linear Model

Source	Degrees of Freedom	Sum of Squares	Mean Squares	Expected Mean Squares
Parts	$I - 1$	S_p	M_p	$\Sigma_r + K \Sigma_{po} + JK \Sigma_p$
Operators	$J - 1$	S_{oo}	M_{oo}	$\Sigma_r + K \Sigma_{po} + IK \Sigma_{oo}$
Parts \times Operators	$(I - 1)(J - 1)$	S_{po}	M_{po}	$\Sigma_r + K \Sigma_{po}$
Error	$IJ(K - 1)$	S_r	M_r	Σ_r
Total	$IKJ - 1$	S_T		

Figure 1: The contour+data plots for the two-dimensional analysis of the hole center data. The dashed line represents the positional tolerance region. The ellipses are the 99% contours from the two-dimensional normal distributions with mean vector 0 covariance matrices: (a) $\hat{\Sigma}_p$, (b) $\hat{\Sigma}_o$, (c) $\hat{\Sigma}_r$, and (d) $\hat{\Sigma}_e$. Tables 1 and 2 contain the numerical summaries that accompany these illustrations.

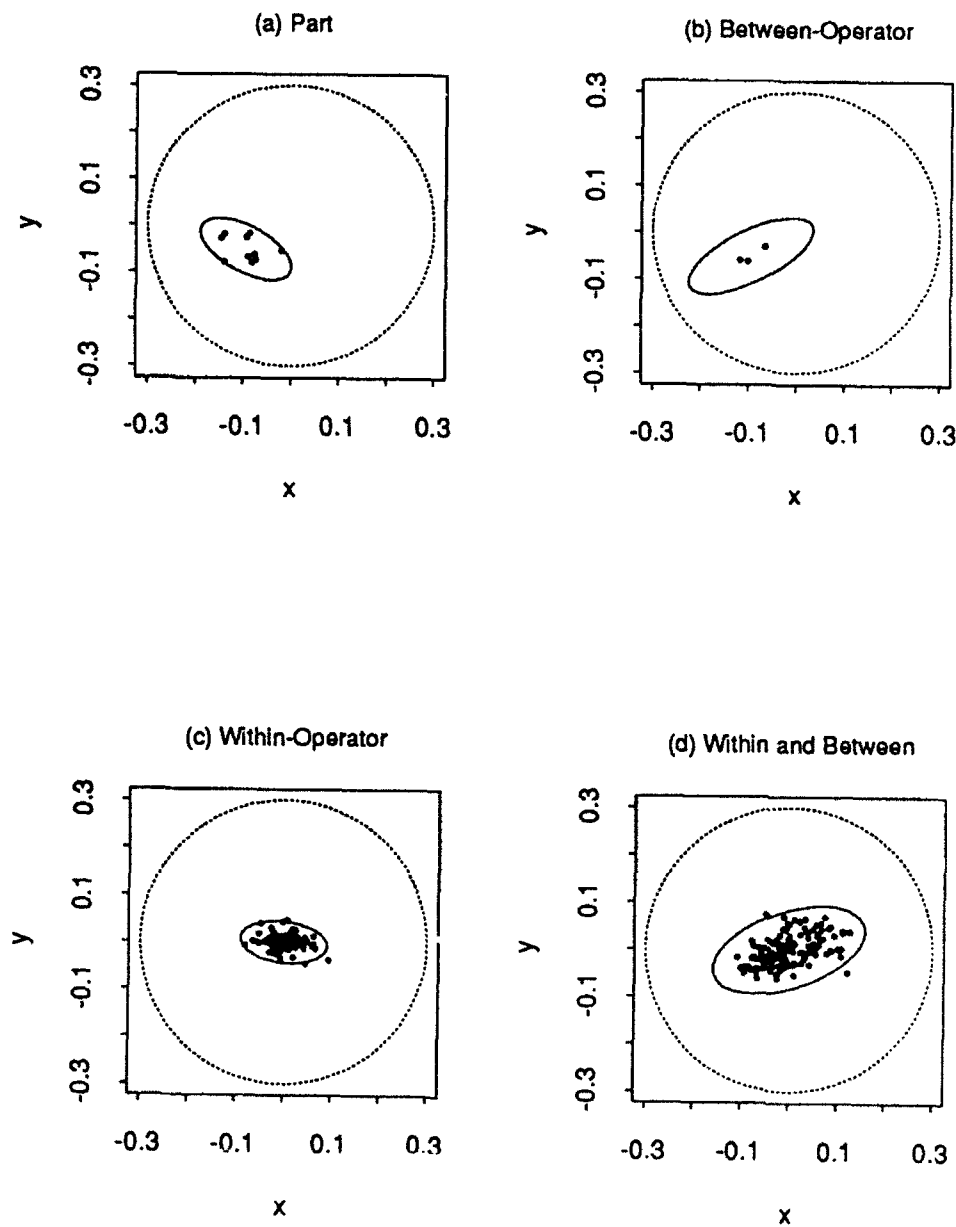


Figure 2: An illustration of the Variance-to-Tolerance ratios comparing Σ_e and the positional tolerance region. Here, the volume ratio is $R_{e(V)}(0.01) = 0.15$ and the length ratio is $R_{e(L)}(0.01) = 0.55$ (See Table 2).

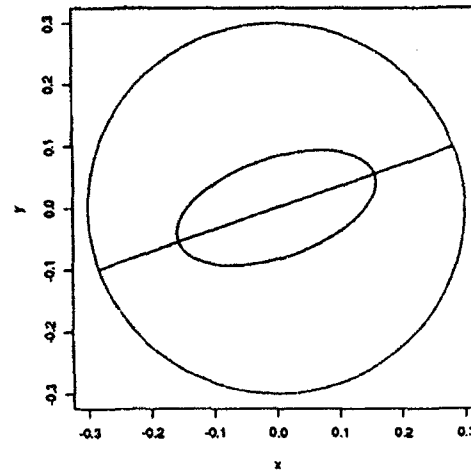
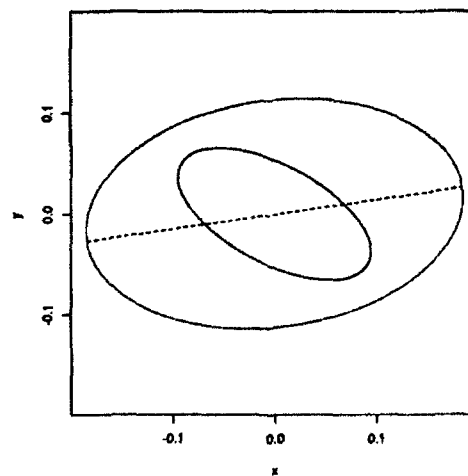


Figure 3: An illustration of the Efficiency ratios comparing Σ_p and $\Sigma_e + \Sigma_p$. Here, the volume ratio is $EFF_{(V)} = 0.24$ and the length ratio is $EFF_{(L)} = 0.37$ (See Table 2).



Exploring Time Series Using Univariate Phase Maps

Edmund L. Russell, III

Advanced Micro Devices MS-524
5900 East Ben White Blvd.
Austin, TX 78741

Abstract: Over the past few years, principally due to the study of fractals, there has been an increased awareness that even relatively simple, completely deterministic, recursive systems may produce time series data that appear to be highly chaotic in nature. Very slight perturbations in the constants controlling these systems can lead to widely differing behavior of the same underlying process.

The univariate phase map, in movie form, can be a highly sensitive detector of shifts or changes in the underlying model of a time series. This paper explores some of the uses of the univariate phase map, in movie form, for exploring time series data and discovering modeling difficulties prior to actually fitting time series models.

1) Introduction

Over the past few years researchers in a variety of disciplines have become highly interested in dynamical systems. Dynamical systems are recursive systems that map back into their own domain. The relationship between dynamical systems and time series analysis becomes obvious when the iterates of a dynamical system are tracked over time. In the language of these systems, these iterates are said to trace out an orbit. Frequently the orbits themselves become an object of study. This is particularly true for orbits that have a large number of iterates.

In some systems we find that the iterates tend to converge to a fixed point. An example of this occurs in the Verhulst population growth model:

$$X_i = r * X_{i-1} * (1 - X_{i-1})$$

where

$$X_i \in (0,1)$$

When the growth parameter, r , is set to 2.99, the iterates converge on 0.665552 approximately.

In other systems we find that the iterates converge to a set of points that recur periodically. An example of this occurs again in the Verhulst model for a growth parameter set between 3 and 3.4495, in which case the iterates eventually cycle between two numbers and the orbit is said to have a period of two. In particular, when r is set to 3.01 the iterates cycle between 0.632849 and 0.699377

In still other dynamical systems the iterates appear to behave randomly but remain close to a set A of points. That is, the orbits that come near the set A tend to remain close to A . This set of points, A , is called an attractor. If the set A is a fractal set, then the set A is called a strange attractor. The converse of an attractor is a repeller, and a fractal repeller is called a strange repeller.

Completely deterministic dynamical systems that contain either a fractal attractor or repeller may exhibit chaotic behavior. Falconer suggests that such systems commonly exhibit the following properties:

- a) the orbit of a member of the attractor or repeller is dense in the attractor or repeller
- b) the periodic points of the iterates of the attractor or repeller are dense in the attractor or repeller
- c) the iterates exhibit sensitive dependence on initial conditions, that is points in the attractor or repeller that are initially close together do not remain close together.

The concept of deterministic chaos is conceptually challenging at first glance. The idea implies that entirely deterministic systems can appear to exhibit random behavior. Even more challenging is that relatively simple deterministic systems can exhibit apparent randomness.

This should not be a surprise to statisticians and computer scientists because of the use of such systems in generating pseudo-random numbers. For example, one of the better systems for generating uniform pseudo-random numbers is the Super-Duper algorithm which has been shown to hold up in five-dimensions:

$$Seed_i = ((Seed_{i-1} * 69069) + 1) \bmod 2^{32}$$

$$U_i = \frac{Seed_i}{2^{32}}$$

For another example we can consider once again the Verhulst population growth model. When the population growth parameter, r , is set to 3.569, the population eventually oscillates about 16 fixed values. However the behavior changes radically when the population growth parameter is set just above 3.56999. In this region, the modeled population can assume an infinite number of values, seemingly at random.

The Quincunx (Galton's Board)

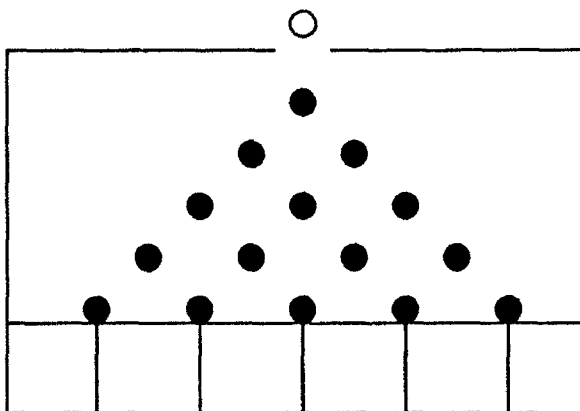


Figure 1.

For a familiar, simple physical example of chaotic behavior, consider the Quincunx in Figure 1. This simple device has been used for years to demonstrate how one can get a normal distribution from a large number of independent Bernoulli trials.

A ball is dropped on the top bar and it bounces down onto the bars below in each case going to the right or left with equal probability. The ball finally comes to rest in one of the bins located below the last row of bars. The balls accumulate in the bins approximating a normal distribution.

However if we believe in the Laws of Physics, there are no independent Bernoulli trials. The entire outcome is fully determined the moment the ball is dropped. So where does the normal distribution come from? The answer is from a field of repellers present on the top bar. This field of repellers, in the limit, appears to be a fractal set.

Real physical dynamical systems that are believed to exhibit chaotic behavior have been discovered. Some examples include: kneading ingredients into a bread dough, turbulent fluid flow, the time between drips of water from a faucet, Brownian motion, the spinning of a water wheel, transmission errors on phone lines, and the beating of a heart.

Time series obviously could be generated from any of the above mentioned physical dynamical systems. These systems can easily be thought of in terms of standard ARIMA models with more or less "noise" tossed in. However, from the point of view of physics, the randomness enters the system in the uncertainty in the initial conditions only. All successive time points are purely deterministic.

II) Univariate Phase Map

One of the traditional plots that is examined in times series analysis is a plot of the data against itself lagged by one or more time periods. This plot is typically presented in the form of a scatter plot of all of the data against itself as seen in Figure 2.

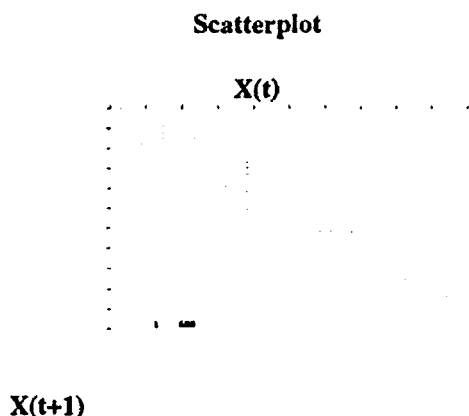


Figure 2: Scatterplot of times series data plotted against itself lagged by one time interval.

It is well known that this type of plot can show that neighboring points in time are autocorrelated. Another name commonly used for such plots is the univariate phase map.

Drawing upon the concept of strange attractors from chaotic systems, it might be inferred that there is a possibility that the univariate phase map might show evidence of such an attractor. The attractor would become evident since orbits of the iterates that are near an attractor tend to remain near the attractor. In fact, phase maps are commonly used to assist in finding attractors of dynamical systems.

One enhancement to the traditional scatter plot form of a univariate phase map is to connect successive points with lines. If a lot of points are plotted, then the resulting plot may become very messy. To reduce the clutter, we can present the connected points in smaller groups. Since we don't know what the optimal breakpoints are for this type of presentation for any given data set, we can present the univariate phase map as a movie.

We would like to know if, for any given autoregressive time series, the univariate phase maps are stable and in what sense. We would also like to know if the univariate phase maps are distinctive enough to allow the investigator to determine which model underlies the time series data.

If there is some stability and the univariate phase map movies of different models appear distinctive, then the univariate phase map movies can aid in uncovering a

number of problems that are encountered in trying to fit autoregressive time series data.

To begin answering these questions, consider an autoregressive process of order 1, an AR(1) process, where we define:

$$X_t = \phi * X_{t-1} + \varepsilon_t$$

Now if we ignore the noise term, ε_t , then we find that the expected slope of the points plotted on the univariate phase map movie for an AR(1) is ϕ . This is because the slope of the line through any two plotted points at times $(t+1, t+2)$, and $(t, t+1)$ is given by:

$$m = \frac{(X_{t+2} - X_{t+1})}{(X_{t+1} - X_t)}$$

which simplifies to ϕ .

This slope is realized if there is essentially no noise in the process. If the noise term needs to be included then there is no ready simplification.

However if we consider an autoregressive process of order 2 defined as:

$$X_t = \phi_1 * X_{t-1} + \phi_2 * X_{t-2} + \varepsilon_t$$

there is no readily interpretable reduction of the slopes of the lines in the univariate phase map movie. For instance, ignoring the noise term as before the slopes of the two joined lines through the plotted points at times $(t+2, t+3)$, $(t+1, t+2)$, and $(t, t+1)$ is given by:

$$m_2 = \frac{(X_{t+3} - X_{t+2})}{(X_{t+2} - X_{t+1})}$$

and

$$m_1 = \frac{(X_{t+2} - X_{t+1})}{(X_{t+1} - X_t)}$$

These slopes simplify to:

$$m_2 = \frac{(\phi_1^2 + \phi_2 - \phi_1)X_{t+1} + \phi_2(\phi_1 - 1)X_t}{(\phi_1 - 1)X_{t+1} + \phi_2 X_t}$$

and

$$m_1 = \frac{(\phi_1 - 1)X_{t+1} + \phi_2 X_t}{X_{t+1} + X_t}$$

It becomes obvious that this becomes worse with even higher order autoregressive processes. And worse yet if the noise terms are included in the equations.

III) Synthetic data examples of Autoregressive Processes

To overcome our lack of awareness of the properties of the univariate phase map movies based on recursive equations in the presence of noise, we turn to simulated processes. In this paper six simulated autoregressive time series are examined, and portions of the univariate phase map movies are presented. Only a small percentage of noise was included in the synthetic data series. The series examined are:

Coefficients negative:

$$X_t = -0.95X_{t-1} + \varepsilon_t$$

$$X_t = -0.95X_{t-1} - 0.95X_{t-2} + \varepsilon_t$$

$$X_t = -0.95X_{t-1} - 0.95X_{t-2} - 0.95X_{t-3} + \varepsilon_t$$

Coefficients positive:

$$X_t = 0.95X_{t-1} + \varepsilon_t$$

$$X_t = 0.05X_{t-1} + 0.95X_{t-2} + \varepsilon_t$$

$$X_t = 0.01X_{t-1} + 0.01X_{t-2} + 0.95X_{t-3} + \varepsilon_t$$

Even with simple casual examination of the univariate phase map movies it is easy to see that there is a distinct difference between autoregressive models with positive coefficients and negative coefficients. In particular, models with positive coefficients tend to have phase map movies that move back and forth on a 45° line from the origin. This can be seen in Figures 3 to 5 but is more evident in movie form.

Autoregressive Time Series AR(1),
Coefficients Positive



Figure 3

Autoregressive Time Series AR(2),
Coefficients Positive

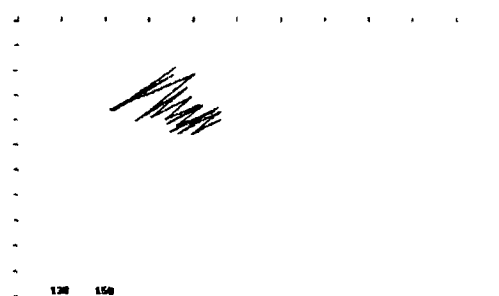


Figure 4

**Autoregressive Time Series AR(3),
Coefficients Positive**

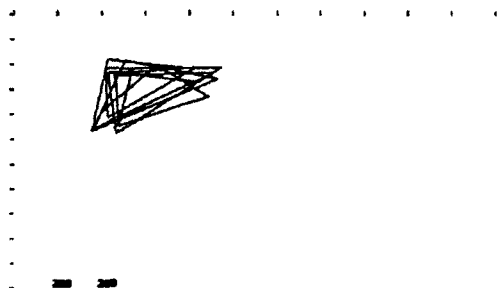


Figure 5

**Autoregressive Time Series: AR(2),
Coefficients Negative**

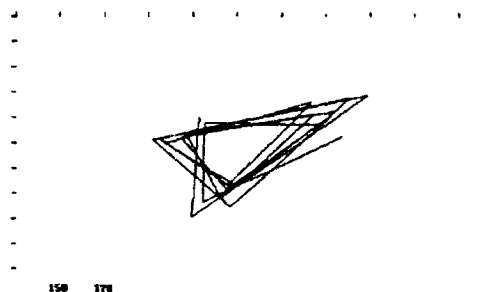


Figure 7

Models with negative coefficients tend to move perpendicular to the same 45° line, see Figures 6 to 8. It is also readily apparent that models of different orders do appear to behave differently.

**Autoregressive Time Series: AR(1),
Coefficients Negative**

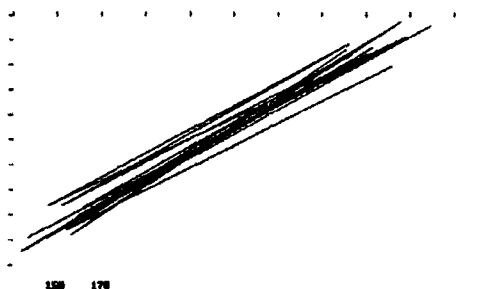


Figure 6

**Autoregressive Time Series: AR(3),
Coefficients Negative**

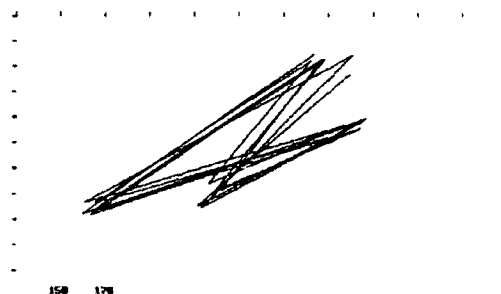


Figure 8

One of the more surprising results is that the observed patterns apparent in the univariate phase map movies are not unconditionally stable. This is most apparent in the higher order models. There appear to be occasions, perhaps corresponding to data points at which the noise term dominated, upon which the pattern of the phase map tends to change to a similar pattern. Compare Figures 8 and 9. In addition there are cases in which the dominant pattern is temporarily abandoned - see Figure 10.

**Autoregressive Time Series: AR(3),
Coefficients Negative**

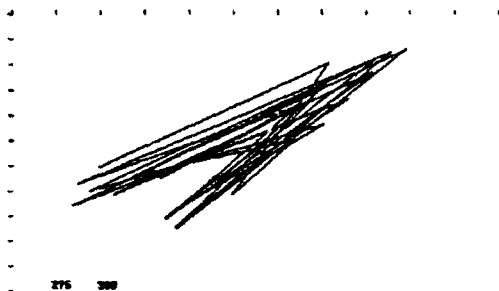


Figure 9: Notice that the pattern has shifted to a pattern similar to that in Figure 8.

**Autoregressive Time Series: AR(3),
Coefficients Negative**

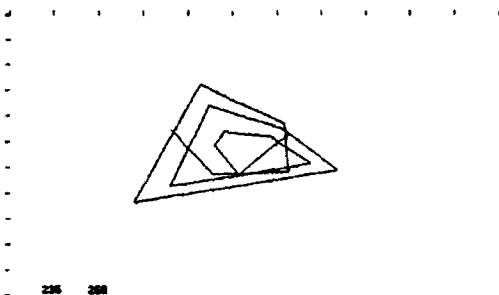


Figure 10: Notice that this pattern is not characteristic for the synthetic data in this data set.

The impact of this lack of stability on the usefulness of the univariate phase map to identify autoregressive time series in noisy data remains to be evaluated.

**IV) Example of uses of the Univariate Phase Map
Movie**

One of the uses that has been made of the univariate phase map movie is to examine data sets for problems. In one case, the author received an electronic copy of a data set taken from the Box, Hunter, Hunter text. Upon running this particular data set through the movie, it was immediately noticeable that the movie completely retraced itself, see Figure 11.

Box, Hunter, Hunter Data Set

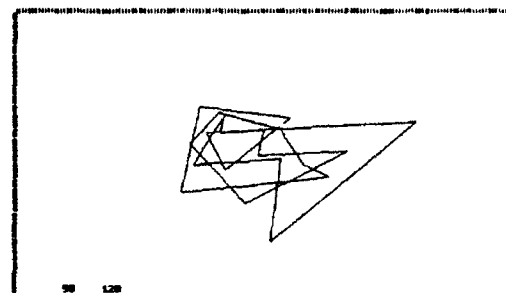


Figure 11: Region of the data set in which the data values were accidentally duplicated. Note that the univariate phase map traces over itself.

Using some of the features in the program, the beginning and end of the retrace was discovered. The data in this section, about 20 observations long, was compared to the data published in the text. It turned that there was a data entry error of 20 duplicate observations. Correcting this error did in fact change the partial autocorrelations for this series.

Another data set which was examined that yielded interesting results is some star magnitude data also provided with another time series analysis package. The full story of this data set is not known to the author at this time, however it is apparent that either the star (if it is real) is behaving in a very peculiar manner from time to time or there has been some manipulation of the data in the data set, see Figures 12 and 13.

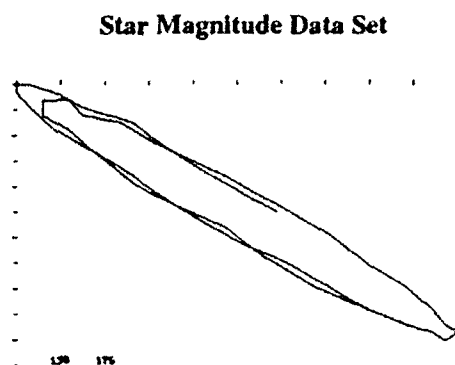


Figure 12: Star Magnitude data as it appears in the univariate phase map movie. This is similar to the majority of the data in the data set.

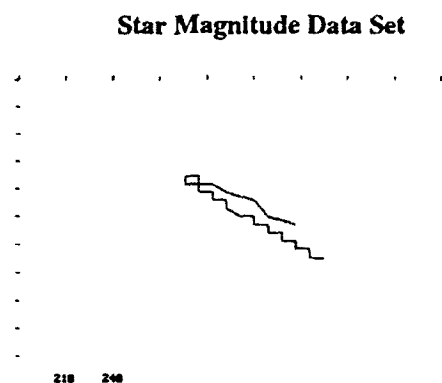


Figure 13: Star magnitude data as it appears in a few small portions of the data set in the univariate phase map movie.

The author does not know how seriously this affects the partial autocorrelations for the data.

Another interesting data set that was studied by the author is some geophysical sonic log data that had been declared as uninterpretable. In this type of data, there is reason to believe that the data contains very little noise. It is also

reasonable to believe that the underlying model changes from time to time and that the geological processes operating in an area will change over time. In addition, processes will repeat from time to time.

What made this data set uninterpretable was that there were no dominant features for the sonic log analysts for "correlation" purposes. Most interpretations from the region are based on well-bore core instead of sonic log data.

Upon comparing obvious dominant patterns in the univariate phase map movie of this data set to an interpretation based on actual well-bore core examination, it was found that whenever similar patterns were evident in the univariate phase map of the sonic log, that the same interpretation appeared based on the core. For an example, see Figures 14 and 15.

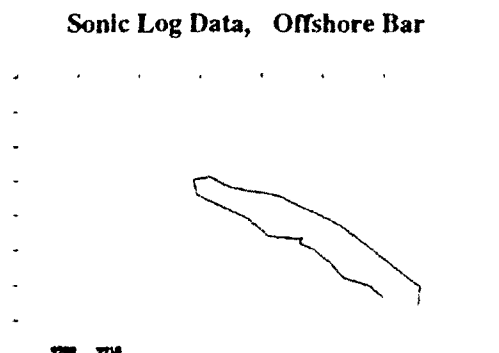


Figure 14: Core corresponding to this portion of the sonic log data was interpreted as coming from an offshore bar. Depth is approximately 9,905 feet.

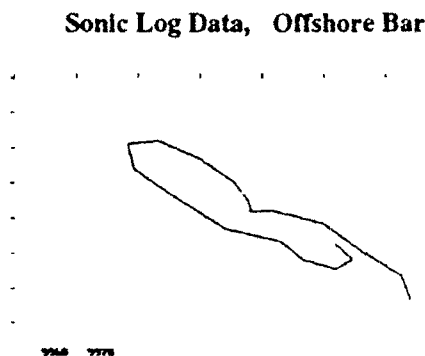


Figure 15: Core corresponding to this portion of the sonic log data was interpreted as coming from an offshore bar. Depth is approximately 9,965 feet.

In addition, in every case that the same interpretation was made based on the core, similar patterns were found in the univariate phase map movie. The patterns that dominated the movie apparently changed each time the geological depositional environment changed. This analysis also held up for nearby wells for which sonic log data were obtainable.

Conclusions

The univariate phase map movie is a useful adjunct tool for examining time series data. More work surely needs to be done before this tool is mature, however preliminary indications are that it can be a highly useful tool in exploring time series data. Some of the potential uses are: detecting major departures from the underlying time series model, detecting cycling among more than one or two models, identification of models and short model fragments in low noise environments, detecting miscoded data as distinguished from data points with large noise components.

The author is also looking at using this technology in Statistical Process Control settings in which: a non-autocorrelated process suddenly becomes autocorrelated, an autocorrelated process suddenly changes underlying models, and an autocorrelated process that has larger than

usual noise components. There is a potential that if the suspect autocorrelation pattern, generally very short, is known from past history, that process intervention can begin at earlier times than previously possible using traditional SPC methodology.

References

- Box, G. E. P., and Jenkins, G. M. (1976), *Time Series Analysis: Forecasting and Control*, Holden Day, San Francisco, California.
- Falconer, K. (1990), *Fractal Geometry, Mathematical Foundations and Applications*, John Wiley and Sons, New York.
- Glass, L., Goldberger, A. L., Courtmancher, M., and Shri A. (1987), "Nonlinear Dynamics, Chaos and Complex Cardiac Arrhythmias," *Dynamical Chaos, Proceedings of Royal Society of London*, Princeton University Press, New Jersey.
- Gleick, James (1985), *Chaos*, Penguin Books, New York.
- Hewett, T. A. (1986), "Fractal Distributions of Reservoir Heterogeneity and Their Influence on Fluid Transport," 6 Annual Technical Conference and Exhibition, Society of Petroleum Engineers.
- May, R. M., F. R. S. (1987), "Chaos and the Dynamics of Biological Populations," *Dynamical Chaos, Proceedings of the Royal Society of London*, Princeton University Press, New Jersey.
- Newton, H. J., (1988), *TIMESLAB: A Time Series Analysis Laboratory*, Wadsworth & Brooks/Cole Publishing Company, Pacific Grove, California.
- Peterson, I. (1987), "Zeroing in on Chaos," *Science News* Volume 131.
- Spiegel, E. A. (1987), "Chaos: A Mixed Metaphor for Turbulence," *Dynamical Chaos, Proceedings of the Royal Society of London*, Princeton University Press, Princeton, New Jersey.
- Stauffer, D. and Stanley, H. E. (1990), *From Newton to Mandelbrot, A Primer in Theoretical Physics*, Springer-Verlag, New York.

von Mises, R. (1981), *Probability, Statistics and Truth*,
Dover Publications, Inc, New York.

Walker, J. R. (1985), "The Pseudo-Random Number
Generator For SYVAC3," Atomic Energy of Canada
Limited Report, AECL-373 .

Weiss, N. O. (1987), "Dynamics of Convection," *Dynamical
Chaos, Proceedings of the Royal Society of London*,
Princeton University Press, New Jersey.

Witten, T. A. and Cates, M. E. (1986), "Tenuous Structures
from Disorderly Growth Processes," *Science*, Volume 232.

Adapting Subregion-Adaptive Integration Software to Problems in Bayesian Inference

Alaattin Erkanli

ISDS

Duke University

Durham, NC 27706

e-mail: ae@durer.isds.duke.edu

Robert E. Kass

Department of Statistics

Carnegie Mellon University

Pittsburgh, PA 15213-3890

e-mail: kass@stat.cmu.edu

Alan Genz

School of EE and Computer Science

Washington State University

Pullman, WA 99164-2752

e-mail: acg@eecs.wsu.edu

Abstract

This paper discusses software, currently under development, for application of subregion-adaptive numerical integration methods to multiparameter Bayesian inference problems. Following Genz and Kass (Proceedings of the 23rd Interface), a parameter transformation is used to transform the domain of integration to a multidimensional unit cube. On the cube, well-tested software developed by Genz and colleagues may be used. We consider (i) the kinds of problems this software should be able to solve, and the inputs to it required of the user and (ii) desirable features the software should have. We also mention current progress in implementation.

KEY WORDS: Gaussian quadrature, multiple integrals, posterior calculations.

1 Introduction

Although a variety of methods have been used to compute integrals arising in Bayesian inference (e.g., Evans, 1992, this volume), their impact on statistical practice is limited by lack of widely available and easily-used software. Especially useful would be a modular collection of FORTRAN subroutines for statistical multiple integration. In this note we outline desirable characteristics of such a collection.

The programs we have in mind should (i) be well-suited to problems users typically wish to solve, (ii) require relatively little information from the user in order to run them, and (iii) be well-tested and documented. In addition, (iv) experienced users should be able to access key components for customization. Ideally, a wide variety of integration strategies should be available. This may increase applicability, but furnishes important checks on results.

Our thinking currently focuses on what we have called (Genz and Kass, 1991) "subregion-adaptive" integration methods (Genz and Malik, 1980; Berntsen, Espelid, Genz, 1991). Well-tested software based on this approach is available (obtainable from the third author through *netlib*), but it is not especially well suited to typical Bayesian integration problems. The subregion-adaptive algorithm assumes the region of integration is an m -dimensional box; it then divides the region into subregions, attempts to identify those in which the integrand varies most, and goes on to further subdivide subregions of high variability. The difficulty in Bayesian applications is that the integrand is usually very highly peaked, so that it is nearly zero throughout most of whatever box one might reasonably pick as the domain of integration. Thus, the subregion-adaptive method inefficiently searches for the subregion where the action is and might even miss it entirely.

On the other hand, it is not hard to fix this problem

lem: the region of peakedness is roughly known once the posterior mode and approximate covariance matrix are found, and these may be used to define transformations to the unit cube. Genz and Kass (1991) used normal distribution functions following an approximate orthonormalization (from the Cholesky decomposition of the approximate covariance matrix), and improvements to this relatively naive approach are certainly possible. Employing any such transformation, the basic steps in using subregion-adaptive methods are (i) compute the modal and modal covariance matrix (i.e., the inverse of the negative Hessian of the log posterior density), (ii) transform to the m -dimensional unit cube, and (iii) apply subregion-adaptive integration. These are also essentially the same steps that need to be followed for a variety of other methods (substituting for (iii) and correspondingly modifying (ii)) including Laplace's method, Gauss-Hermite integration, and Monte Carlo importance sampling.

The guiding heuristic here, of course, is that under regularity conditions and for large samples, posterior distributions are approximately normal; in many applications the approximating normal distribution is itself of interest. On the other hand, in some applications posterior distributions are clearly not close to normal and alternative methods are needed. Subregion-adaptive integration is often applicable in irregular as well as regular cases, but in illustrating our discussion here we presume we would be in the "well-behaved" situation.

2 What We Would Like to Compute

Several items are essential for Bayesian data analysis.

- The posterior mode and the Hessian of the log posterior density, which furnish the modal normal approximation to the posterior.
- Expectations and variances of components of the parameter vector.
- The marginal densities of components of the parameter vector.
- Interval probabilities for components of the parameter vector.

In addition, it is highly desirable to be able to compute the following.

- The marginal density of the data.

- The probability content of a joint region, especially the content of elliptical regions based on the normal approximation.
- For real functions of the parameter vector, such as log odds ratio, modes and modal approximate standard deviations, expectations, variances, marginal densities, and marginal probabilities.

A class of models deserving special attention, Kass and Steffey (1989) have called *Conditionally dependent Hierarchical Models*, consists of two-stage hierarchical models that arise in parametric empirical applications. These hierarchical models result in likelihood functions that are themselves products of integrals so that posterior calculations involve iterated integrals. Where possible, this special structure should be exploited.

3 Desirable Features of the Software

We would like the final software to be organized as a portable package of FORTRAN subroutines with the following features.

- Both "easy" versions, which require relatively little user input, and "detailed" versions, which allow full control.

For example, in computing expectations and variances a relatively simple routine might have the following inputs and outputs.

inputs:

log_posterior - a user-defined function

data - a user-defined subroutine that returns an array with dimensions specified by data_dimensions

data_dimensions - a two-integer vector giving number of rows and columns in array returned by the subroutine data

mode - a vector: the posterior mode

modal_covariance - an array: the posterior covariance matrix

parameter_dimension - an integer giving the dimension of the

```

parameter vector
outputs:
  expectation - a vector
  covariance_matrix - an array
  posterior_constant - a number: the
    normalizing constant for the
    posterior

```

On the other hand, a more detailed version might include inputs involving alternative functions to be included, types of transformations to be used, the integration method, desired accuracy, and limits on the amount of work required to obtain the results. Additional outputs could, for instance, include assessments of accuracy. In addition, an especially simple version could incorporate evaluation of the mode, rather than leaving the user to apply two separate subroutines.

- The ability to call individual modules, depending on what quantities are needed.

This is important partly because users may wish to focus on some particular part of the calculation, possibly repetitively, and also because it assists users in introducing variations on existing methods.

- The ability to retain intermediate calculations for subsequent use (to improve efficiency).

In many methods of integration this could be important. One may want to store a collection of parameter values and integrand evaluations in order to re-use it following perturbation of the likelihood or prior or to calculate expectations of additional functions.

- The availability of other computational methods.

One typically begins with the modal approximation, so subroutines to make that convenient in practice are needed (e.g., a subroutine that applies the delta method would be useful). If subregion-adaptive integration is regarded as an improvement on the modal approximation, then Laplace's method would be helpful as a check. Other methods could be added on as well.

- The ability to handle "conditionally independent" hierarchical models efficiently.

This is especially important because of the wealth of repeated-measures applications.

4 Concluding Remarks

We have been investigating alternative transformations to the unit cube. An initial prototypical driver program has been written by the first author to exemplify modules for (i) maximization (ii) Hessian computation (iii) transformation and (iv) subregion adaptive integration for the computation of expectations and variances. Some revision, testing, and documentation of this program is necessary before we will be able to make it available. Once this is done, we would like to go to the much larger project we have outlined here. We are hoping to receive advice and suggestions from our colleagues concerning the design of this collection of programs.

This work was partially supported by NSF grant DMS 9008125 and NIH grant 1-54037.

References

- Berntsen J., Espelid T.O., and Genz, A. (1991). An adaptive algorithm for the approximate calculation of multiple integrals. *ACM Trans. Math. Software*, pp. 437-451.
- Evans, M. (1992) Some integration strategies for problems in statistical inference. In this volume.
- Genz, A. and Kass, R.E. (1991). An application of region adaptive integration to a Bayesian inference problem, *Computing Science and Statistics: Proceedings of the 23rd Symposium on the Interface*, Ed. E. Keramidas, Interface Foundation of America, pp. 441-444.
- Genz, A. and Malik, A.A. (1980). An adaptive algorithm for numerical integration over an dimensional rectangular region. *J. Comp. Appl. Math.* 6, pp. 295-302.
- Kass, R. E. and Steffey, D. (1989). Approximate Bayesian inference in conditionally independent hierarchical models (parametric empirical Bayes models). *J. Am. Stat. Assoc.* 84, pp. 717-726.

COMPUTING DOUBLY NON-CENTRAL *t* PROBABILITIES

J. Michael Hardin and Karan P. Singh
 University of Alabama at Birmingham,
 University Station, Birmingham, Alabama 35294

ABSTRACT

Krishan (1968) and Bulgren and Amos (1968) gave representations of the doubly non-central *t* distribution and its computations. Their techniques involve both double summations and integrals. Recently Kocherlakota and Kocherlakota (1991) have given an alternative representation of the distribution. Their techniques involve the singly noncentral *t* distribution and uses the IMSL subroutine DTNDF. In this paper we propose simpler methods for computing the doubly non-central *t* Probabilities. Our techniques do not involve the subroutine DTNDF and have better efficiency and accuracy. These techniques have been implemented in a computer program written in ANSI 77 FORTRAN.

KEY WORDS: Doubly and singly non central *t* distribution; Poisson distribution function; series representation; incomplete beta integral; error bound.

1 INTRODUCTION

Let z denote a normal random variable with mean δ and unit variance, and let Y be an independent chi-square random variable with n degrees of freedom and non-centrality parameter λ . Then the random variable T defined as a ratio $z\sqrt{n}/\sqrt{Y}$ follows a doubly noncentral *t*-distribution with n degrees of freedom and non-centrality parameter δ and λ . When $\lambda = 0$ and $\delta \neq 0$, T reduces to the usual (singly) noncentral *t* random variable given by Amos (1964), Resnikoff and Lieberman (1957) and others. Recently Singh et. al. (1992) have provided a simple method to compute the tail probabilities of the singly noncentral *t* distribution. When $\delta = 0$ and $\lambda \neq 0$, T follows

the noncentral *t* distribution given by Marakathavalli (1954); and, when $\delta = \lambda = 0$, becomes Student's central *t* random variable. Several applications of the doubly noncentral distribution are given in Robins (1948), Patr (1955), Krishan (1968) and others. Krishan (1968) and Bulgren and Amos (1968) have given series representations using functions including incomplete beta, hypergeometric (Gauss and Confluent) and modified Bessel functions. However, their techniques involve series of double summations as well as integrals. Calculations of the double infinite series of integrals involving three parameters n , δ , and λ are not simple due to problems both in terms of the time and the termination of the summations.

Kocherlakota and Kocherlakota (1991) have given an alternative representation for the doubly noncentral *t* distribution involving the single noncentral *t* distribution. Their technique uses the IMSL subroutine DTNDF to evaluate the singly noncentral *t* distribution function. This representation reduces the double infinite series of integrals to a single infinite series. Singh et. al. (1992) have proposed an alternative technique for small, moderate and large values of (n, δ) , as well as a computer program, DNONCT, which implements the procedure. In comparing results, it has been shown that DNONCT does a better job in terms of speed and accuracy, as DNONCT can provide a given, desired accuracy. It is noted that this technique only works for integer or half integer degrees of freedom. This is not a serious limitation, however, since fractional degrees of freedom occur rarely in applied statistics (Satterthwaite 1946; Gayler and Hopper, 1969).

In this short note, we propose a method for evaluating the doubly noncentral *t* distribution incorporating the technique given by Singh et. al. (1992) for evaluating the singly

noncentral t probabilities into the expression recently developed by Kocherlakota and Kocherlakota (1991). We compare our results for selected values of (t, n, δ, λ) with those obtained by using the expression of Kocherlakota and Kocherlakota (1991) and the IMSL subroutine DTNDF. The IMSL subroutine DTNDF is based on approximate formulae whereas our subroutine is based on exact expressions for the incomplete beta (Singh and Relyea, 1992). Table 1 shows that our method does better in terms of accuracy and is a good alternative for IMSL. A FORTRAN computer program has been developed to implement this method. The program can be easily converted to any computer supporting ANSI FORTRAN 77 and it is available upon request.

2 COMPUTATION OF DOUBLY NON-CENTRAL t PROBABILITIES

Let Z , Y and T be the random variables defined in section 1. Then following Kocherlakota and Kocherlakota (1991)

$$P(T \leq t | n, \delta, \lambda) = \sum_{k=0}^{\infty} q_k(\lambda/2) \int_0^{\infty} \psi_{n+2k}(y) \Phi[-\delta + t(y/n)^{1/2}] dy \quad (2.1)$$

where

$$q_k(\theta) = \frac{e^{-\theta} \theta^k}{k!} \quad (\text{Poisson weight}), \quad \Phi(\cdot)$$

is the standard normal distribution function and

$$\psi(y) = \frac{e^{-y/n} y^{n/2-1}}{2^{n/2} \sqrt{n/2}}$$

(the pdf of the central chi-square random variable with n degrees of freedom).

By differentiating the cumulative distribution (2.1), we easily obtain the pdf of T . Krishnan (1968) has given a different representation for the cdf of T . His expression for $t > 0$ is given by

$$P(T \leq t | n, \delta, \lambda) = \frac{1}{2} \sum_{k=0}^{\infty} \sum_{j=0}^{\infty} P(\lambda, k) P(\delta^2, \frac{j}{2}) [(-1)^j + I_w\left(\frac{j+1}{2}, \frac{n+2k}{2}\right)] \quad (2.2)$$

and for $t < 0$,

$$P(T \leq -t | n, \delta, \lambda) = \frac{1}{2} \sum_{k=0}^{\infty} \sum_{j=0}^{\infty} P(\lambda, k) P(\delta^2, \frac{j}{2}) (-1)^j [1 - I_w\left(\frac{j+1}{2}, \frac{n+2k}{2}\right)] \quad (2)$$

where

$$P(\theta, k) = q_k(\theta), \quad w = \frac{t^2}{(t^2 + n)}, \quad \text{and}$$

$$I_w(p, q) = \frac{1}{B(p, q)} \int_0^w x^{p-1} (1-x)^{q-1} dx$$

(the incomplete beta function) with $B(p, q)$, complete beta function.

Similar expressions are given by Bulgai and Amos (1968). Note that expressions (2.2) (2.3) involve double infinite series and present problems with respect to terminating summations in evaluating these expressions.

Kocherlakota and Kocherlakota use an interesting trick to make expression (2.1) Poisson-weighted sum of singly noncentral cdf's. Let $\lambda = 0$ for the random variable T . The distribution function of T is then given by

$$P(T \leq t | n, \delta, 0) = \int_0^{\infty} \psi_n(y) \Phi[-\delta + t(y/n)^{1/2}] dy \quad (2)$$

Using (2.4) in (2.1) and adjusting the degrees of freedom for the singly noncentral t distribution function in (2.4), (2.1) can be rewritten as Poisson-weighted sum of singly noncentral distribution functions with $n + 2k$ degrees of freedom and non-centrality parameter δ . The expression then can be written as

$$P(T \leq t | n, \delta, \lambda) = \sum_{k=0}^{\infty} q_k(\lambda/2) P[T \leq t' | n + 2k, \delta] \quad (2)$$

where $t' = t \left(\frac{n+2k}{n} \right)^{1/2}$ and $q_k(\theta) = e^{-\theta} \theta^k / k!$

Following Lenth (1989) and Singh et al. (1991) the singly noncentral distribution function, $P[T \leq t' | n + 2k, \delta]$, can be rewritten as

$$P(T \leq t' | n + 2k, \delta) = \sum_{j=0}^{\infty} [P_{1j} I_1(j + .5, k + \frac{n}{2}) + P_{2j} I_2(j + 1, k + \frac{n}{2})] \quad (2)$$

where

$$x = \frac{(t')^2}{[(t')^2 + n + 2k]},$$

$$P_{1j} = \frac{1}{2} e^{-\delta^2/2} \frac{(\delta^2/2)^j}{j!},$$

$$P_{2j} = \frac{1}{2} \delta P_{1j} \frac{j!}{[\sqrt{2} \Gamma(i+1.5)]},$$

and

$$E_m < 2 \left(1 - \sum_{j=0}^m P_{1j} \right) I_x \left(m + \frac{3}{2}, k + \frac{n}{2} \right) \quad (2.7)$$

Note that E_m is an upper error bound.

A computationally simpler technique for evaluating (2.6) for integer or half-integer degrees of freedom is given in Singh et. al. (1992). Based on this technique, a documented FORTRAN computer program (which can be run on an 386 IBM-PC compatible with Math Co-processor (33MH)) has been developed. A sketch of the details for recursive formulae are given below.

If the degrees of freedom, n , is an even integer, then the incomplete beta integrals $I_x(j+1/2, k+n/2)$ and $I_x(j+1, k+n/2)$ can be evaluated exactly for each j , for a given k , as follows:

$$I_x(a, b) = \begin{cases} 1 - (1-x)^b \sum_{i=0}^{a-1} \left(\prod_{\alpha=1}^i \frac{b+\alpha-1}{\alpha} \right) x^i & \text{if } a \text{ is an integer} \\ x^a \sum_{i=0}^{b-1} \left(\prod_{\alpha=1}^i \frac{a+\alpha-1}{\alpha} \right) (1-x)^i & \text{if } b \text{ is an integer} \end{cases}$$

If n is an odd integer, then $I_x(j+1/2, k+n/2)$ is exactly evaluated by

$$I_x(a, b) = \frac{1}{2} + \frac{2}{\pi} [x(1-x)]^{1/2} D1(D2-D3) - \frac{1}{\pi} \sin^{-1}(1-2x) \quad (2.8)$$

where

$$d_1 = \left[\sum_{i=1}^{b-3} \left(\frac{i}{i-.5} x \right) \right]$$

$$D_2 = \left[\sum_{i=1}^{b-3} \left(\prod_{\alpha=1}^{i-1} \frac{a+\alpha-.5}{\alpha+.5} \right) (1-x)^{i-1} \right], \text{ and}$$

$$D_3 = \sum_{i=1}^{a-3} \left(\prod_{\alpha=1}^{i-1} \frac{\alpha}{\alpha+1} \right) x^{i-1}$$

Now, $I_x(j+1/2, n/2)$, $I_x(j+1, k+n/2)$, P_{1j} and P_{2j} each j and for fixed k can be computed using the following recursive formulae:

$$A_j = A_{j-1} - C_{j-1} x^{a+j-1} (1-x)^b$$

$$P_{1j} = \frac{1}{2} \frac{\delta^2}{2j} P_{1j-1}, \text{ and}$$

$$P_{2j} = \frac{1}{2} \frac{\delta^2}{2j+1} P_{2j-1}$$

where

$$A_0 = I_x(a, b), \quad A_j = I_x(a+j, b),$$

$$P_{10} = \frac{1}{2} e^{-\delta^2/2}, \quad P_{20} = \delta P_{10} \sqrt{(2/\pi)}$$

with $a = .5$ or 1 and $b = k + n/2$.

Also, note that

$$C_{j-1} = \begin{cases} \prod_{i=1}^{a+j-1} \frac{b+i-1}{i}, & a \text{ is an integer} \\ \prod_{i=1}^b \frac{a+j+i-1}{i}, & b \text{ is an integer} \\ \frac{1}{(a+j-1)B(a+j-1, b)} & \text{otherwise} \end{cases}$$

The algorithm of Pike and Hill (1966) can be used to evaluate C_{j-1} when neither a nor b are integers.

Using the above results we compute noncentral t probability, $P(T \leq t' | n+2k, \delta)$ each k . We terminate the sum in (2.5) when the absolute difference between two consecutive terms is less than 10^{-15} . We can get the probability by subtracting the cumulative probability from 1.0. When t is negative we use the following equality (Krishnan, 1968; Bulgr

1968):

$$P(T \leq -t|n, \delta, \lambda) = 1 - P(T \leq t|n, -\delta, \lambda) \quad (2.10)$$

The results and comparisons of this method for selected values of (t, n, δ, λ) are given in Table 1.

3 CONCLUSION

In this short note we have proposed a simple method to compute the doubly noncentral t probabilities. This method gives both greater accuracy and higher efficiency. At present we are developing an upper bound to terminate the infinite sum in (2.5). These new results will be presented in a future communication.

REFERENCES

- Amos, D.E. (1964). Representations of the Central and Non-central t -Distribution. *Biometrika*, 51, 451-458.
- Bulgren, W.G. and Amos, D.E. (1968). A Note on the Representations of the Doubly Non-central t -Distribution. *J. Amer. Statist. Assoc.*, 64, 1013-1019.
- Gaylor, D.W., and Hopper, F.N. (1969). Estimating the Degrees of Freedom for Linear Combinations of Mean Squares by Satterthwaite's Formula. *Technometrics*, 11, 691-706.
- Krishnan, M. (1968). Series Representations of the Doubly Non-Central t -Distribution. *J. Amer. Statist. Assoc.*, 63, 1004-1012.
- Kocherlakota, K., and Kocherlakota, S. (1991). On the Doubly Non-Central t -Distribution. *Commun. Statist.-Simula.*, 20, 23-31.
- Lenth, R.V. (1987). Computing Distribution of the Non-Central t -Distribution, *Appl. Statist.*, 38, 185-189.
- Marakatharalli, N. (1954). The Distribution of t , and its Application, *Journal of the Madras University*, 24B, 251-272.
- Patnaik, P.B. (1955). Hypotheses concerning the means of observations in normal samples, *Sankhya*, 15, 343-372.
- Resnikoff, G.J., and Lieberman, G.J. (1955). *Tables of the Non-Central t -Distribution* Stanford University Press.
- Robbins, H. (1948). The Distribution of Student t when the Population Means are Unequal. *Annals of Math. Statist.*, 19, 406-610.
- Satterthwaite, F.E. (1946). An Approximate Distribution of Estimates of Variance Components, *Biometrics Bulletin*, 2, 110-114.
- Singh, K.P., Relyea, G.E., and Bartolucci, A. (1991). On the Tail Probabilities of the Non-Central t -Distribution, *Computational Statistics*, 7, 67-80.
- Singh, K.P., and Relyea, G.E. (1995). Computation of Non-Central F Probabilities: Computer Program, *Computational Statistics & Data Analysis*, 13, 95-102.

Table 1
Cumulative Probability $F(t|\delta, \lambda, n)$ of
Doubly Noncentral t -distribution

		$\lambda = -2$					
		0		4		8	
n	t	(1)*	(2)**	(1)	(2)	(1)	(2)
2	-4	.94327	.0090	.68737	.0498	.44106	.0890
	-2	.57952	.0635	.23375	.1095	.09030	.1284
	0	.09175	.0652	.02107	.0764	.00494	.0790
	2	.00127	.0014	.00022	.0127	.00004	.0167
	4	.00000	.0000	.00000	.0001	.00000	.0002
5	-4	.96147	.537.01	.87170	.537.01	.75337	.537.00
	-2	.53629	.61.239	.32549	.61.242	.19146	.61.244
	0	.05097	.1104	.01837	.1269	.00690	.1328
	2	.00032	.61.251	.00008	.61.261	.00002	.61.268
	4	.00000	.537.01	.00000	.537.01	.00000	.537.01
10	-4	.96914	.0047	.99371	.0100	.88628	.0171
	-2	.51903	.0481	.39114	.0586	.29018	.0669
	0	.3669	.0739	.01898	.0768	.00994	.0782
	2	.00013	.0286	.00005	.0385	.00002	.0464
	4	.00000	.0003	.00000	.0006	.00000	.0010
15	-4	.97183	.368.13	.95105	.368.13	.92442	.368.13
	-2	.51289	.21.327	.42131	.21.333	.34293	.21.338
	0	.03197	.1611	.01960	.1674	.01211	.1712
	2	.00009	.21.342	.00004	.21.351	.00002	.21.359
	4	.00000	.368.13	.00000	.368.14	.00000	.368.14
20	-4	.97318	.0035	.95875	.0054	.94094	.0078
	-2	.50974	.0116	.43847	.0350	.37494	.0379
	0	.02963	.0751	.02008	.0766	.01369	.0776
	2	.00007	.0466	.00004	.0546	.00002	.0618
	4	.00000	.0007	.00000	.0011	.00000	.0016

Note: * (1) denotes the probability calculating from the proper
 ** (2) denotes the absolute error = |(1) - (2)| * 10^{-4}

Table 1 (continued)

Cumulative Probabilities $F(t|\delta, \lambda, n)$ of Doubly Noncentral t-distribution

λ	t = -1				t = 0				t = 1				t = 2			
	(1)	(2)	(1)	(2)	(1)	(2)	(1)	(2)	(1)	(2)	(1)	(2)	(1)	(2)		
1	.99721	.0004	.97696	.0036	.99860	.0095	1.00000	.0000	1.00000	.0001	1.00000	.0003	1.00000	.0000	1.00000	
2	.84395	.0213	.63094	.0472	.45799	.0585	.99014	.0030	.99901	.0165	.99973	.0720	.99973	.0034	.99996	
3	.21132	.0461	.08807	.0657	.03871	.0736	.78868	.0461	.91195	.0657	.96129	.0766	.90825	.0032	.99796	
4	.00386	.0030	.00099	.0165	.00027	.0030	.15605	.0213	.36906	.0472	.54201	.0585	.42048	.0035	.90980	
5	.00000	.0000	.00000	.0001	.00000	.0003	.00279	.0004	.07204	.0036	.06140	.0095	.05673	.0090	.55894	
6	.99815	.22135	.99430	.22135	.98803	.22135	1.00000	.22135	1.00000	.22135	1.00000	.22135	1.00000	.537.01	.537.01	
7	.84086	.35.059	.74774	.35.047	.65825	.35.054	.99768	.35.042	.99900	.35.053	.99914	.35.064	.99958	.61.251	.99998	
8	.18161	.0940	.11401	.1284	.07358	.1490	.81839	.0940	.88599	.1284	.97642	.1690	.94971	.1104	.99310	
9	.00232	.35.042	.00100	.35.053	.00046	.35.064	.15814	.35.039	.25226	.35.047	.34175	.35.6	.46371	.61.259	.80854	
10	.00000	.221.35	.00000	.221.35	.00185	.221.35	.00185	.221.35	.00570	.221.35	.01197	.221.35	.03853	.537.01	.24663	
11	.99841	.0002	.99707	.0003	.99523	.0005	1.00000	.0001	1.00000	.0002	1.00000	.0003	1.00000	.0003	1.00000	
12	.84148	.0097	.79346	.0105	.74609	.0112	.99818	.0153	.99891	.0223	.99933	.0292	.99937	.0286	.99998	
13	.17045	.0526	.13107	.0589	.10203	.0635	.82955	.0526	.84893	.0589	.89797	.0635	.96331	.0739	.99006	
14	.00182	.0153	.00109	.0223	.00067	.0292	.15852	.0097	.20654	.0105	.25391	.0112	.48997	.0481	.70982	
15	.00000	.0001	.00000	.0002	.00000	.0003	.00159	.0002	.00293	.0003	.00477	.0005	.02086	.0047	.11372	
16	.99850	.301.60	.99771	.301.60	.99672	.301.60	1.00000	.301.60	1.00000	.301.60	1.00000	.301.60	1.00000	.368.13	1.00000	
17	.84141	.10.758	.80926	.10.763	.77720	.10.767	.99834	.10.762	.99885	.10.768	.99919	.10.773	.99991	.21.342	.99996	
18	.16659	.1288	.13869	.1430	.11626	.1545	.83141	.1288	.86131	.1430	.88374	.1545	.96803	.1611	.98789	
19	.00166	.10.762	.00115	.10.768	.00081	.10.773	.15859	.10.758	.19074	.10.763	.22270	.10.767	.48711	.21.327	.57869	
20	.00000	.301.60	.00000	.301.60	.00150	.301.60	.00150	.301.60	.00729	.301.60	.00328	.301.60	.02817	.368.13	.07558	
21	.99854	.0001	.99796	.0001	.99733	.0002	1.00000	.0001	1.00000	.0002	1.00000	.0003	1.00000	.0007	1.00000	
22	.84138	.0049	.81723	.0045	.79316	.0041	.99842	.0202	.99882	.0245	.99910	.0238	.99993	.0466	.99996	
23	.16463	.0535	.14300	.0570	.12477	.0599	.83537	.0535	.85700	.0570	.87523	.0599	.97037	.0751	.97992	
24	.00158	.0202	.00118	.0245	.00090	.0288	.15862	.0049	.18277	.0045	.20684	.0041	.49726	.0316	.67506	
25	.00000	.0001	.00000	.0002	.00000	.0003	.00146	.0001	.00202	.0001	.00267	.0002	.02682	.0035	.05006	

ASYMPTOTIC TEST STATISTICS FOR INFINITELY DIVISIBLE DISCRETE DISTRIBUTIONS I

John J. Hsieh, University of Toronto

ABSTRACT This article establishes the conditions for the weak convergence to normality of compound Poisson distributions and obtains approximate test statistics for the general compound Poisson distribution and three specific infinitely divisible discrete distributions.

1. INTRODUCTION

The theoretical importance of the Poisson distribution and its frequent occurrences in the modeling of real-world phenomena arise from the fact that the *stationary independent increments and infinite divisibility properties* (abbreviated as SIIDP) of the (homogeneous) Poisson process ensures preservation of the Poisson distribution under filtration, thinning, superposition and decomposition. Furthermore, the SIIDP of the Poisson process carries over to *compound* and *generalized* Poisson processes. This underlies the common practice of employing compound or generalized Poisson distributions to improve the goodness of fit of the ordinary Poisson distribution when the orderliness condition of the Poisson process is no longer tenable. The SIIDP also ensures that the Poisson process and under certain conditions, its generalizations will tend to Gaussian Processes as limits as a quantity of information (such as the number of processes under superposition or the sample size or the indexing parameter value) increases, the Gaussian limit so obtained being the only continuous process that preserves SIIDP.

Modeling of the ordinary and compound Poisson processes have found wide applications in various scientific disciplines and real life situations, and much work has been done in this area (see, e.g., Haight (1967) and Snyder and Miller (1991) and references cited therein). Our main concern in this article is in the other area where more work is needed, namely, the use of these models in statistical tests of significance and determination of confidence regions. In particular, we are interested in making statistical inference of the *infinitely divisible discrete distributions* (IDDD). To do so, it would be convenient (computationally at least) to obtain simple expressions of the asymptotically normal test statistics for these distributions.

The objectives of the present article are (i) to establish the conditions for the weak convergence to normality of compound Poisson distributions and (ii) to use simple transformations to derive *asymptotic unit normal statistics*

(AUNS) for compound Poisson distributions in general for three IDDDs in particular. In Part II of the paper will compare various transformed statistics so obtained the Poisson and three discrete compound Poisson distributions with the ones in use with respect to accuracy of approximation to normality.

In Section 2 we describe the convergence theorem will be used to generate the AUNSSs. In Section 3 we use theorem and two simple transformations to obtain the statistics for Poisson distribution, which are Z_1 , Z_2 and given in (10), (13) and (16), respectively. Section 4 demonstrates the convergence in law of compound Poisson variates to normality and obtain the conditions for such convergence. Section 5 derives the three general test statistics for compound Poisson distribution, namely, Z_1^* , Z_2^* and given in (28), (29) and (30), which are then specialized to the corresponding statistics for two IDDDs: Z_4 , Z_5 and for negative binomial distribution given in (36) and Z_7 and Z_9 for doubly Poisson (Neyman's type A) distribution given in (41). The AUNSSs presented in this article are generated on the basis of two criteria: simplicity and accuracy. If accuracy alone were the sole concern, one could do better using asymptotic expansion methods, such as the Cornish-Fisher inversion of direct Edgeworth expansion and saddlepoint expansion (see, e.g., Barndorff-Nielsen and Cox (1989)) to improve the existing standard test statistics. But this would not directly produce the kind of simple statistics we seek. Instead we shall in Part II of the paper use these asymptotic expansions to assist in assessing the accuracy of the test statistics generated in this article.

2. A USEFUL CONVERGENCE THEOREM

The AUNSSs to be generated in this article are based on the concept of infinite divisibility. Essentially, they are deduced from the fact that averages of random samples from Poisson and, under certain conditions, from compound Poisson distributions have asymptotic normal distributions and so smooth transformations of sample means of Poisson and compound Poisson distributions are also approximately normally distributed. By suitable choice of the transformations and proper application of convergence theorems, we endeavor to produce simple test statistics that hopefully will accomplish good approximations to normality. We describe below the convergence theorem for approximating distributions of functions of sample averages of independent

identically distributed (i.i.d.) random variables. To meet the infinite divisibility requirement and to generate the simple statistics we want, we have restructured somewhat the more general theorems available in Rao (1973, Ch.6) or Serfling (1980, Ch.3).

Theorem 1: Let $\{U_n\}$, $n=1,2,\dots$, be a sequence of i.i.d. random variables with mean $E(U_n) = \mu^*$ and positive, finite

variance $\text{Var}(U_n) = \sigma^2(\mu^*)$ and let $\bar{U}_n = \sum_{i=1}^n U_i / n$ be the average of the first n observations. If

$$(U_n - \mu^*)/\sigma(\mu^*) \xrightarrow{L} Z, \quad (1)$$

where Z has the unit normal distribution and \xrightarrow{L} signifies convergence in law (in distribution), and (i). if g is a continuous function of a real variable having a nonzero derivative $g'(\mu^*)$ at μ^* , then

$$\sqrt{n}(g(\bar{U}_n) - g(\mu^*)) / [g'(\mu^*)\sigma(\mu^*)] \xrightarrow{L} Z. \quad (2)$$

(ii). If g' and σ are also continuous, then

$$\sqrt{n}(g(\bar{U}_n) - g(\mu^*)) / [g'(\bar{U}_n)\sigma(\bar{U}_n)] \xrightarrow{L} Z. \quad (3)$$

Proof: The theorem is proved in three steps: *First*, (1) implies (4):

$$\sqrt{n}(\bar{U}_n - \mu^*)/\sigma(\mu^*) \xrightarrow{L} Z. \quad (4)$$

This follows from the preservation of normality under linear transformation and the use of continuity theorem. *Second*, (4) implies (2): This is proved by Taylor expansion of the transformed statistic $g(\bar{U}_n)$ around μ^* up to the first derivative term followed by the use of (4) and Slutsky's theorem to obtain the convergence in probability of \bar{U}_n to μ^* . *Third*, (4) implies (3): This follows from the first and second steps upon using the fact that convergence in law (equivalently, convergence in probability to a fixed parameter) holds for continuous functions, so that

$$\bar{U}_n \xrightarrow{P} \mu^* \text{ implies } g'(\bar{U}_n)\sigma(\bar{U}_n) \xrightarrow{P} g'(\mu^*)\sigma(\mu^*). \quad (5)$$

To use Theorem 1 to generate the AUNSS one must first establish (1) and then choose a suitable transformation g for expressions (2) and (3). Two simple transforms that serve to improve normal approximation of data are the square root transformation and the log transformation. The former also serves to stabilize the variance for the transformation of the compound Poisson (and hence Poisson) variate while the latter serves to remove positive skewness and equalize variances (the log transformation is stronger than the square root transformation). We therefore expect that use of these two transforms for the function g in (2) and (3) would produce AUNSS that are in general more accurate than the standardized normal variate Z_1 below. Other theorems and techniques can also be derived to produce more asymptotic statistics. However, statistics deduced from other methods

either do not have a simple form (such as the Peizer-F (1968) statistics) or do not provide as good approximation as those deduced from Theorem 1 (see last paragraph Section 3).

Because of its importance and the fact that it requires no condition for convergence to normality, we shall treat the special case of Poisson distribution in Section 3 ahead of the general results in Sections 4 and 5.

3. ASYMPTOTIC STATISTICS FOR POISSON DISTRIBUTION

Suppose the random variable U_n in Theorem 1 has Poisson distribution so that $E(U_n) = \text{Var}(U_n) = \mu^* > 0$ for n . We must first establish the asymptotic normality of U_n .

Let $\phi_{U_n}(s) = E[e^{sU_n}]$ be the characteristic function (c.f.) of U_n . We write $W_n = (U_n - \mu^*)/\sqrt{\mu^*}$ where U_n has Poisson distribution with

$$\phi_{U_n}(s) = E[e^{sU_n}] = \exp[\mu^*(e^s - 1)].$$

Then

$$\begin{aligned} \ln \phi_{W_n}(s; \mu^*) &= \mu^* [e^{s/\sqrt{\mu^*}} - 1] - is\sqrt{\mu^*} \\ &= -s^2/2 + O(1/\sqrt{\mu^*}), \end{aligned}$$

which tends to $-s^2/2$ (which is the log c.f. of the unit normal distribution) at rate $1/\sqrt{\mu^*}$ (a rather slow rate) as $\mu^* \rightarrow \infty$. Then, in view of Theorem 1, expressions (2) and (3) hold with $\sigma^2(\mu^*) = \mu^*$. Now let

$$Y = n\bar{U}_n \quad (6)$$

be the sum of n i.i.d. Poisson random variables. It follows from (6) that Y is a Poisson random variable and that

$$\nu = n\mu^* \quad (7)$$

is the mean and the variance of Y . It then follows from (7) that, as $\nu \rightarrow \infty$,

$$Z_1 = (Y - \nu)/\sqrt{\nu} \xrightarrow{L} Z, \quad (8)$$

which says that the standardized Poisson variate converges in law to the unit normal distribution as its mean (variance) $\rightarrow \infty$. Note that expression (10) can also be obtained by applying the linear transformation (11) to (2) followed by the use of (8) and (9):

$$g(x) = \sqrt{n} x. \quad (9)$$

Clearly, Poisson processes $\{U_n(t)\}$ and $\{Y(t)\}$ obtain when U_n and Y are also indexed by a parameter t (which may be interpreted as time) and in this case $\mu^* = E[U_n(t)]$, $\text{Var}[U_n(t)] = \lambda t$ and $\nu = E[Y(t)] = \text{Var}[Y(t)] = n\lambda t$, for $\lambda > 0$, for $n=1, 2, \dots$. Thus, a Poisson random variate (process) can be regarded as the sum (superposition) of Poisson random variables (processes) with independent identical distribution whose mean (which equals variance) is n times the common mean (equaling the common variance).

of the component random variables (processes), $n=1, 2, \dots$. That is to say, the Poisson variable (process) is infinitely divisible.

Next, let

$$g(x) = \sqrt{x}. \quad (12)$$

Substituting (12) in (2) and using (8) and (9), we obtain

$$Z_2 = 2(\sqrt{Y} - \sqrt{v}) \xrightarrow{d} Z. \quad (13)$$

Note that substitution of (12) in (3) instead of (2) would produce the same result. Using the fact that Z_1 is asymptotically normal, the convergence in law of Z_2 to the unit normal distribution as $v \rightarrow \infty$ can alternatively be proved as follows:

Alternative Proof of (13):

$$\begin{aligned} \{Z_2 \leq x\} &= \{2(\sqrt{Y} - \sqrt{v}) \leq x\} \rightarrow \{\sqrt{Y} \leq x/2 + \sqrt{v}\} \\ &\rightarrow \{Y \leq x^2/4 + x\sqrt{v} + v\} \rightarrow \{Z_1 \leq x + x^2/4\sqrt{v}\}. \end{aligned} \quad (14)$$

Thus, Z_2 and Z_1 will have the same asymptotic unit normal distribution as $v \rightarrow \infty$. ■

Finally, let

$$g(x) = \ln(x). \quad (15)$$

Substitute (15) in (3) and use (8) and (9) to yield

$$Z_3 = \sqrt{Y} \ln(Y/v) \xrightarrow{d} Z. \quad (16)$$

Alternative proof of (16):

Using the relations in (14) one establishes below the asymptotic equivalency of the two events $\{Z_2 \leq x\}$ and $\{Z_3 \leq x\}$ as $v \rightarrow \infty$.

$$\begin{aligned} \{Z_2 \leq x\} &\rightarrow \{Z_3 \leq (x/2 + \sqrt{v}) \ln(1 + x/\sqrt{v} + x^2/4v)\} \\ &= \left(\frac{x}{2} + \sqrt{v}\right) \left[\frac{x}{\sqrt{v}} + \frac{x^2}{4v} - \frac{1}{2} \left(\frac{x}{\sqrt{v}} + \frac{x^2}{4v} \right)^2 + O(v^{-3}) \right] = x + O(v^{-1}). \end{aligned} \quad (17)$$

This result together with the fact that Z_2 has asymptotic normal distribution provides an alternative proof of the asymptotic normality of Z_3 . ■

By choosing appropriate fractions as exponents, power transforms other than square root can be used to improve normal approximations. However, the resulting formulas are not as convenient as the square root. Simple statistics other than Z_1 , Z_2 and Z_3 obtained above (such as $Z_1' = (Y-v)/\sqrt{Y}$ and $Z_3' = \sqrt{v} \ln(Y/v)$) can also be derived from the convergence theorem given in Section 2. However, they are not nearly as accurate as the three obtained here. By a slight modification of (2) or (3) one can also obtain slightly more complicated statistics such as $Z_2'' = 2(\sqrt{Y} - \sqrt{v})\sqrt{(Y/v)}$ and $Z_3'' = (Y/\sqrt{v}) \ln(Y/v)$. But again these statistics are not more accurate than the three obtained here.

4. ASYMPTOTIC NORMALITY OF THE COMPOUND POISSON DISTRIBUTION

The distribution of the sum of a random number of d. random variables is known as the *compound distribution*, so called because in the case of discrete distributions, probability generating function (p.g.f.) of the sum obtained by compounding (or compositing) that of random number with that of the i.i.d. random variable. Specifically, let N be a counting random variable and X discrete or continuous random variable such that

$$Y = \sum_{i=1}^N X_i = \sum_{i=1}^{\infty} I(i \leq N) X_i. \quad (18)$$

If N has Poisson distribution with mean μ , say, and X_1, \dots are i.i.d. copies of X which is also independent of N , then Y is said to have a *compound Poisson distribution* -- a *generalized Poisson distribution* when X is discrete. terms of the *compound* or *generalized Poisson process*, (18) becomes

$$Y(t) = \sum_{i=1}^{N(t)} X_i = \sum_{i=1}^{\infty} I(t_i \leq t) X_i, \quad (18a)$$

where the t_i 's are the times at which jumps occur at Poisson rate λ . In (18a) X_i may be interpreted as the size or mark of the i th jump at each jump time t_i of the N jumps, $N(t)$ being a poisson process with mean $\mu = \lambda t$, and $Y(t)$ as the cumulative sum of the X_i up to time t . Just as the Poisson process, the intervals (t_i, t_{i+1}) between jump times in compound Poisson processes are independent exponential variables. When X has the degenerate distribution with $\Pr(X=1)=1$, then $Y=N$, almost sure and the compound Poisson distribution (process) specializes to the Poisson distribution (process).

From the definitions above it is not difficult to see connection of compound Poisson processes with the SIII. In fact (see Loève (1977) and Lukacs (1970)), every infinitely divisible (decomposable) discrete distribution is a compound Poisson distribution and every infinitely divisible distribution (i.e., any distribution the n th root of whose characteristic function is a characteristic function, $n=2,3,\dots$) is the complete limit of compound Poisson distributions (de Finetti's theorem). Furthermore, a mixed distribution becomes a compound distribution if its parameter being mixed is the number of convolutions of distribution with itself. When the number of convolutions has a Poisson distribution, then the resulting convolution mixed distribution is the compound Poisson distribution. These explain why many well-known distributions can be derived as compound Poisson distributions and so render new interpretations and why compound Poisson processes have such wide theoretical and applied interest.

We now establish the conditions for the weak convergence of compound Poisson random variables to normality; rederive a general recursive formula for computing the discrete compound Poisson probability distribution. Note v

that in this article μ and μ_0 always stands for the mean of the Poisson random variable N in (18) or $N(t)$ in (18a).

Proposition 1: For the compound Poisson distribution and process defined in the first paragraph of this section, suppose moments of all orders for the distribution of X exist, then the compound Poisson variable Y converges in law to normal distribution with asymptotic mean $\mu E(X)$ and variance $\mu E(X^2)$, as $\mu \rightarrow \infty$, provided $E(X^r)/E^{r/2}(X^2)$ is bounded for $r \geq 3$.

Proof: For the compound Poisson variable Y defined by (18), the c.f. is, upon using (6) and the compound relation $\phi_Y(s) = \phi_N(\ln \phi_X(s)/i)$, given by

$$\begin{aligned}\phi_Y(s) &= E[e^{isY}] = E[E[\exp(is \sum_{j=1}^N X_j) | N]] \\ &= e^{\mu E[e^{isX} - 1]} = e^{\mu[\phi_X(s) - 1]},\end{aligned}\quad (19)$$

where X has finite moments of all orders and has the same distribution as each X_j in (18), so that

$$\ln \phi_Y(s) = \mu E[e^{isX} - 1] = \mu \left[isE(X) - \frac{1}{2}s^2 E(X^2) + \sum_{r=3}^{\infty} \frac{i^r}{r!} s^r E(X^r) \right] \quad (20)$$

$$\text{for all } s \text{ such that } \frac{1}{|s|} > \limsup_{k \rightarrow \infty} (|E(X^k)|/|k|)^{1/k}.$$

From (20) we immediately obtain a simple general expression for the r th cumulant (semi-invariant) of Y :

$$\begin{aligned}\kappa_r &= \mu E(X^r), \quad r=1,2,3,\dots, \text{ so that} \\ v &= E(Y) = \kappa_1 = \mu E(X), \quad \sigma^2 = \text{Var}(Y) = \kappa_2 = \mu E(X^2).\end{aligned}\quad (21)$$

Now, write the c.f. of $(Y-v)/\sigma$ in terms of that of Y and expand the exponential function in it and use (20) and (21), to yield

$$\begin{aligned}\ln \phi_{\frac{Y-v}{\sigma}}(s) &= -is \frac{v}{\sigma} + \ln \phi_Y(s/\sigma) = -\frac{1}{2}s^2 \\ &+ \sum_{r=3}^{\infty} \frac{i^r}{r!} \frac{E(X^r)}{\mu^{r/2-1} E^{r/2}(X^2)} \rightarrow -s^2/2, \text{ as } \mu \rightarrow \infty \text{ provided} \\ &E(X^r)/E^{r/2}(X^2) \text{ remains bounded for } r=3,4,\dots\end{aligned}\quad (22)$$

This completes the proof of the Proposition. By substituting $\mu = \lambda t$ into the proof above we also have the following Corollary: The compound Poisson process $Y(t)$ is asymptotically normal with (asymptotic) mean $\lambda t E(X)$ and variance $\lambda t E(X^2)$, as $t \rightarrow \infty$, provided $\lambda^{r/2-1} E(X^r)/E^{r/2}(X^2)$ is bounded for $r \geq 3$. ■

Note that by employing instead of (19) the Lévy-Khintchine's canonical representation for infinitely divisible characteristic functions (see Steutel(1973) and Loève(1977)), the condition for the complete convergence of the general infinite divisible distribution to normality can also be obtained. Note also that for the special case of Poisson process, $E(X^r) = 1$ for all r , so that all cumulants in (21) reduce to the Poisson mean μ (or λt) and (22) also reduces to (7).

We can also compare the asymptotic cumulants κ of the standardized variate $Y^* = (Y - \mu E(X))/\sqrt{\mu E(X^2)}$ as $\mu \rightarrow \infty$ with those of the standard normal to assess the asymptotic normality of the compound Poisson variate Y . From (20) obtain:

$$\kappa_1^* = 0, \quad \kappa_2^* = 1, \quad \kappa_r^* = \mu^{1-r/2} E(X^r)/E^{r/2}(X^2), \quad r=2,3,\dots$$

Thus as $\mu \rightarrow \infty$ all cumulants of orders greater than 2 tend to zero while the first and second cumulants remain as 0 and 1, respectively, provided $E(X^r)/E^{r/2}(X^2)$ is bounded for $r \geq 3$. These are precisely the cumulants of the standard normal distribution. Thus, under the condition just stated, compound Poisson distribution does in fact tend to normal form as the Poisson mean $\mu \rightarrow \infty$. Furthermore has kurtosis $\kappa_4/\kappa_2^2 \geq$ skewness $\kappa_3/\kappa_2^{3/2}$. (This follows from (21) and Cauchy-Schwarz inequality $E(X^4)E(X^2) \geq E^2(X^3)$). When the compound Poisson distribution specializes to Poisson distribution, the Cauchy-Schwarz inequality becomes an equality and hence $\kappa_4/\kappa_2^2 = \kappa_3^2/\kappa_2^3$. This shows that for the same degree of long-tailedness, a compound Poisson distribution is generally more symmetrical than Poisson distribution specialized from it.

For application of Proposition 1, the condition on moments of X is translated into the conditions on parameters associated with the distribution of X . For Poisson distribution the condition of Proposition 1 is always satisfied. For most compound Poisson distributions, the condition is satisfied only on a restricted parameter space but there are compound Poisson distributions for which the condition is not satisfiable. Thus, not all compound Poisson distributions (as well, not all IDDDs) have asymptotic normal distributions. This will be illustrated in Section 5.

From the infinitely divisible c.f. (19) (which clearly satisfies the infinite divisibility condition $\phi^{1/n}(s; \mu) = \phi(s; \mu/n)$ for $n=2, 3, \dots$), we can now derive the probability distribution of any infinitely divisible discrete variable Y in terms of the probability distribution of X (For continuous distributions the inversion of (19) is generally more difficult). If X is nonnegative integer-valued, so is Y (see (18)), and in this case it helps to replace the two c.f. ϕ_Y and $\phi_X(s)$ in (19) by the corresponding p.g.f. $\pi_Y(u)$ and $\pi_X(u)$ simply by replacing $\exp(is)$ in (19) by u , yielding

$$\pi_Y(u) = e^{\mu(\pi_X(u)-1)} = \prod_{k=1}^{\infty} \exp(\mu P(X=k) \{u^k - 1\}), \quad |u| \leq 1 \quad (23)$$

Eq.(23) expresses an infinitely divisible discrete process as the superposition of independent decomposed discrete compound Poisson processes of jump size k , each associated with a Poisson process with mean $\mu P(X=k)$, $k=1, 2, \dots$. Now, differentiate (23) k times with respect to u and then substitute $u=0$ in the resulting equations, to obtain

$$\pi_Y^{(k)}(0) = \mu \sum_{j=0}^{k-1} \binom{k-1}{j} \pi_X^{(k-j)}(0) \pi_Y^{(j)}(0)$$

which yields the following recursive formulas for computing the discrete probability distribution of Y .

$$P(Y=k) = \frac{\mu}{k} \sum_{j=0}^{k-1} (k-j)P(X=k-j)P(Y=j), \quad k=1,2,\dots \quad (24)$$

with $P(Y=0) = e^{-\mu[1-P(X=0)]}$.

The recurrence relation (24) seems to have been first obtained by Adelson (1966). Note that (24) includes the Poisson probability distribution as a special case (simply substitute $P(X=1)=1$ and $P(X=j)=0$ for $j \neq 1$ in (24).) Eq. (24) will be employed to obtain probability distributions of two discrete compound Poisson variables in Sections 5.1 and 5.2 and in Part II of the paper.

5. ASYMPTOTIC STATISTICS FOR COMPOUND POISSON DISTRIBUTIONS

We now use Proposition 1 and Theorem 1 to generate AUNs for compound Poisson distributions. Suppose the random variable U_n in Theorem 1 has compound Poisson distribution with mean $E(U_n) \equiv \mu^* = \mu_0 E(X)$ for all n , so that we have from the above definition of μ^* and (21),

$$\text{Var}(U_n) = \sigma^2(\mu^*) = \mu^* E(X^2)/E(X). \quad (25)$$

Note that (25) is positive as $E(X^2)$ and the Poisson mean μ_0 are. Then, under the assumed condition of Proposition 1, U_n has asymptotic normal distribution as $\mu_0 \rightarrow \infty$, with (asymptotic) mean μ^* and variance given by (25). In view of this result and Theorem 1, expressions (2) and (3) hold. Now, let

$$Y = n\bar{U}_n. \quad (26)$$

Then Y is the sum of n i.i.d. compound Poisson random variables U_1, \dots, U_n and consequently Y has a compound Poisson distribution with mean

$$v = n\mu^* = \mu E(X), \quad (27)$$

where $\mu = n\mu_0$, and variance $v E(X^2)/E(X)$ (This easily follows from (19) and (21)). It then follows that

$$Z_1^* \equiv \frac{Y-v}{\sqrt{v E(X^2)/E(X)}} \xrightarrow{L} Z. \quad (28)$$

Now, put μ^* and \bar{U}_n in the square root transformation (12) and then substitute the resulting expressions together with (25) into (2), followed by the use of (26) and (27), to obtain

$$Z_2^* \equiv 2(\sqrt{Y}-\sqrt{v})\sqrt{E(X)/E(X^2)} \xrightarrow{L} Z. \quad (29)$$

Similarly, an application of (25, with μ^* replaced by \bar{U}_n , (26), (27) and the log transformation (15) to (3) leads to

$$Z_3^* \equiv \sqrt{YE(X)/E(X^2)} \ln(Y/v) \xrightarrow{L} Z. \quad (30)$$

Note that all asymptotic distributions with $\mu \rightarrow \infty$ in (28)-(30) hold under the assumed condition of Proposition

1 (for Poisson distribution, $E(X)=1$ and so $v=\mu$). It also that just as was done in (29) and (30), the expressions (28) can also be obtained by using the linear transformation (11). The alternative proofs given in Section 3 of the asymptotic normality of Z_2^* and Z_3^* here as well.

The expressions for Z_1^* , Z_2^* and Z_3^* given in (28)-(30) are the three asymptotically unit normal test statistics in general forms for compound Poisson distributions, be continuous or discrete. They reduce to the expressions for Z_2 and Z_3 given in (10), (13) and (16) in Section 3 for Poisson distribution when $E(X)=E(X^2)=1$ is substituted in (28), (29) and (30). We next obtain their specific expressions for two discrete compound Poisson distributions, negative binomial distribution (Section 5.1) and the discrete Poisson distribution (Section 5.2).

5.1 Negative Binomial Distribution

Negative binomial distribution serves as an example of the compound Poisson distribution that can not be equated to Poisson mixing. In the definition of compound Poisson distributions given in (18) suppose X has logarithmic series distribution with parameter p ($0 < p \leq 1$) so that

$$P(X=k) = -(1-p)^k/(k \ln p), \quad k=1,2,\dots$$

Then a substitution of the p.g.f. of X

$$\pi_X(u) = \ln(1-(1-p)u)/\ln p$$

into (23) yields the p.g.f. for Y as

$$\pi_Y(u) = (p/[1-(1-p)u])^{-\mu/\ln p},$$

which shows that Y has negative binomial distribution with parameters $-\mu/\ln p$ and p . Alternatively, one may substitute the logarithmic series probability function (31) into (24) $P(X=0)=0$ to yield the probability function for Y :

$$P(Y=k) = \frac{\Gamma(-\mu/\ln p + k)}{k! \Gamma(-\mu/\ln p)} p^{-\mu/\ln p} (1-p)^k, \quad k=0,1,\dots$$

Moments about the origin of X is easily obtained from (32). With $E(X) = (p-1)/(p \ln p)$ and $E(X^2) = (p-1)/(p^2 \ln p)$ obtain

$$E(X)/E(X^2) = p.$$

Upon substituting (35) into (28)-(30), the expressions for three general statistics Z_1^* , Z_2^* and Z_3^* reduce to

$$Z_1^* \equiv (Y-v)/\sqrt{v p}, \quad Z_2^* \equiv 2\sqrt{p}(\sqrt{Y}-\sqrt{v})$$

$$\text{and} \quad Z_3^* \equiv \sqrt{p Y} \ln(Y/v),$$

where $v = -\mu(1-p)/(p \ln p)$ is obtained either from (32) from (32) and (27). To check the condition of Proposition 1 we use again the calculated moments of X and find $E(X)/E^2(X^2)$ is bounded for all p in $(0,1]$ and that

$$\lim_{p \rightarrow 0} E(X^r)/E^{r/2}(X^2) = \infty, \text{ for } r=3,4,\dots \quad (37)$$

Thus, under the condition $0 < p \leq 1$, the three statistics in (36) for the negative binomial distribution all converge in law to the unit normal distribution, as μ (and hence v) $\rightarrow \infty$.

The above implies that Z_4 , Z_5 and Z_6 may be used as approximate unit normal test statistics for large μ provided p is not too small (or when μ is replaced by λt , for large t provided both λ and p are not close to 0).

From (33) or (34) we observe that by setting $\mu = -\ln p$, the negative binomial distribution of Y reduces to the geometric distribution with $\phi_Y(u) = p/(1-(1-p)u)$ or $P(Y=k) = p(1-p)^k$. In this case, as μ tends to ∞ , p must tend to 0, which implies that $E(X^r)/E^{r/2}(X^2) \rightarrow \infty$ for all $r \geq 3$ (see (37)). Therefore, the condition of Proposition 1 cannot be satisfied and the geometric distribution are not accurately approximated by a normal distribution.

The geometric distribution provides an example of infinitely divisible discrete (and hence compound Poisson) distributions that is not asymptotically normal. That the geometric distribution is a compound Poisson distribution can be seen from the definition given in (18) where N has Poisson distribution with mean $-\ln p$ and X has logarithmic series distribution with probability function given by (31).

5.2 Doubly Poisson Distribution

From the second paragraph of Section 4, it is clear that any distribution which has a parameter in the exponent of its c.f. can be used as a compounding distribution for the compound Poisson distribution that is equatable to Poisson mixing. We study an example of this kind in this section.

When the random variable X in (18) has a Poisson distribution with parameter Θ , say, then the resulting compound Poisson variable Y has *Neyman's Type A* or *doubly Poisson* distribution, which, in view of (6) and (21), has p.g.f.

$$\pi_Y(u) = \exp\{\mu[\exp\{\Theta(u-1)\}-1]\}. \quad (38)$$

The probability function of Y is obtained from (24) upon using $P(X=0)=e^{-\Theta}$ as

$$\begin{aligned} P(Y=k) &= \frac{\mu\Theta}{k} e^{-\Theta} \sum_{j=0}^{k-1} \frac{\Theta^j}{j!} P(Y=k-1-j) \\ &= \frac{e^{-\mu}}{k!} \sum_{j=1}^{\infty} e^{-\Theta j} (\Theta j)^k \frac{\mu^j}{j!}. \end{aligned} \quad (39)$$

From the last expression of (39) one can also interpret the distribution of Y as a mixture of two Poisson distributions - by mixing the conditional random variable $Y|N$ which has a Poisson distribution with mean ΘN with the random variable N which has a Poisson distribution with mean μ .

With $E(X) = \text{Var}(X) = \Theta$, we obtain

$$E(X)/E(X^2) = 1+\Theta. \quad (40)$$

Upon substituting (40) into (28)-(30), the expressions for three general statistics Z_1^* , Z_2^* and Z_3^* reduce to

$$\begin{aligned} Z_1^* &= (Y-v)/\sqrt{v/(1+\Theta)}, \quad Z_2^* = 2\sqrt{(1+\Theta)}(\sqrt{Y}-\sqrt{v}) \\ \text{and} \quad Z_3^* &= \sqrt{(1+\Theta)}Y \ln(Y/v), \end{aligned}$$

where $v = \mu\Theta$ is obtained either from (38) or from (6) (27). To check the condition of Proposition 1 we use moments of X calculated from (6) and find $E(X^r)/E^{r/2}(X^2)$ is bounded for all positive Θ and that

$$\lim_{\Theta \rightarrow 0} E(X^r)/E^{r/2}(X^2) = \infty, \text{ for } r=3,4,\dots \quad (41)$$

Therefore, under the condition $\Theta > 0$, the three statistics (40) for the doubly Poisson distribution all converge in law to the unit normal distribution, as μ (and hence v) $\rightarrow \infty$.

The above implies that Z_7 , Z_8 and Z_9 may be used as approximate unit normal test statistics for large μ provided Θ is not too small (or when μ is replaced by λt , for large t provided both λ and Θ are not close to 0).

Note that the doubly Poisson distribution can have more than one mode and depart considerably from normal form when Θ is large and μ small, but this does not affect its good approximation to normality when μ is large while Θ is not large but not too small.

REFERENCES

- Adelson, R. M. (1966), "Compound Poisson Distribution: *Operational Research Quarterly*, 17, 73-75.
- Barndorff-Nielsen, O. E., and Cox, D. R. (1989), *Asymptotic Techniques for Use in Statistics*, London: Chapman and Hall.
- Haight, F. A. (1967), *Handbook of the Poisson Distribution*, New York: Wiley.
- Loève, M. (1977), *Probability Theory I*, 4th ed. New York: Springer-Verlag.
- Lukacs, E. (1970), *Characteristic Functions*, 2nd ed. London: Charles Griffin.
- Peizer, D. B., and Pratt, J. W. (1968), "A Normal Approximation for Binomial, F, Beta, and Other Common Related Tail Probabilities I & II," *JASA*, 63, 1416-1483.
- Rao, C. R. (1973), *Linear Statistical Inference and Its Applications*, 2nd ed. New York: Wiley.
- Serfling, R. J. (1980), *Approximation Theorems of Mathematical Statistics*, New York: Wiley.
- Snyder, D. L., and Miller, M. I. (1991), *Random Processes in Time and Space*, New York: Springer-Verlag.
- Steutel, F. W. (1973), "Some Recent Results in Infinite Divisibility," *Stochastic Processes Appl.* 1, 125-143.

Multiple Component, Linear Conditional Probability Models for Finite State Markov Processes

Jeffrey P. Benedict and William F. Szewczyk
National Security Agency
Ft. George G. Meade, Maryland

Abstract. In this paper, a generalization of Raftery's linear conditional probability (LCP) model (a finite state mixture transition distribution model) is developed – the multiple component, linear conditional probability (MUCLICOP) model. It retains the parsimony and interpretability of the LCP model while expanding the range of finite state processes that can be approximated. First the MUCLICOP model is introduced and some of its properties described. Several statistical issues are then discussed including a backfitting algorithm for parameter estimation. The paper concludes with an example that showcases the MUCLICOP methodology.

1. Introduction

Finite state Markov processes are an important tool in applied probability. Unfortunately, an M -state, L^{th} -order Markov process has, in general, $(M-1)M^L$ parameters, making modeling using such a process a difficult or impossible task for M 's and L 's of even moderate size. In this paper, a generalization of the linear conditional probability (LCP) model of Raftery is developed – the multiple component, linear conditional probability (MUCLICOP) model – that retains the parsimony and interpretability of the LCP model while expanding the range of finite state processes that can be approximated.

Let X_1, X_2, \dots be an M -state process. (Without loss of generality, the state space for the X 's will be taken to be $\{0, 1, \dots, M-1\}$.) The LCP model for $\{X_t\}$ is

$$\begin{aligned} p(i_0 | i_1, \dots, i_L) \\ &= P(X_t = i_0 | X_{t-1} = i_1, \dots, X_{t-L} = i_L) \\ &= \sum_{k=1}^L \lambda_k q(i_0 | i_k) \end{aligned}$$

where $i_0, i_1, \dots, i_L \in \{0, 1, \dots, M-1\}$, $Q = (q(i | j))_{M \times M}$ is a

column stochastic matrix, and $\sum \lambda_k = 1$. Of course, $Q\lambda = (\lambda_1, \dots, \lambda_L)$ must generate $p(i_0 | i_1, \dots, i_L)$'s ≥ 0 . This constraint is satisfied if $\lambda \geq 0$ but useful LCP models are excluded by requiring nonnegativity of the λ 's (see Ta and Raftery(1991)). When $\lambda \geq 0$, however, the LCP model has a ready interpretation: λ_k = the probability that the X -process returns to a condition similar to that it was in k steps in the past; $q(i | j)$ = the probability that the process then transitions from state j to state i . One can think of such an LCP model as an explicit formulation of a hidden Markov model. The LCP model can be extended and modified in several ways (see Section 4 of Raftery(1985)); MUCLICOP models are one such extension.

The remainder of this paper is organized as follows: In Section 2, the MUCLICOP model is introduced and several of its properties are described. Model fitting by maximum likelihood is discussed in Section 3 where a backfitting algorithm is presented. An example showcasing the machinery of Section 3 is presented in Section 4. In the example, MUCLICOP models are used to model a raster image. This example clearly demonstrates the ability of MUCLICOP models to handle Markovity at very large lags. Some final comments and observations appear in Section 5.

2. MUCLICOP Models

The MUCLICOP model for $\{X_t\}$,

$$\begin{aligned} p(i_0 | i_1, \dots, i_L) \\ &= P(X_t = i_0 | X_{t-1} = i_1, \dots, X_{t-L} = i_L) \\ &= \sum_{k=1}^K \gamma_k \sum_{l=1}^L \lambda_{kl} q_k(i_0 | i_l), \end{aligned}$$

is a mixture of LCP models, i.e., $\sum \gamma_k = 1$ and $\gamma_k \geq 0$. As with LCP models, the number of parameters in a MUCLICOP model grows linearly with L , the number of independent parameters being $K - 1 + K[(L - 1) + M(M - 1)]$.

MUCLICOP model can be interpreted in almost the same way as the LCP model. Unlike the LCP model, however, the MUCLICOP model can accommodate qualitatively different behavior of the dependence between the present and the past at different lags.

Let $Q = \sum \gamma_k Q_k$ and let π denote a stationary distribution of Q , i.e., $Q\pi = \pi$ with $\pi_1 + \dots + \pi_M = 1$. Assume that $\{X_t\}$ follows a MUCLICOP model.

Theorem 1.

Suppose $p(i_0 | i_1, \dots, i_L) > 0$ for all i_0, i_1, \dots, i_L in $\{0, \dots, M-1\}$ ($\Rightarrow Q > 0 \Rightarrow \pi$ exists, is positive and is the stationary distribution of the Markov chain having Q as its transition matrix.) Then

$$\lim_{t \rightarrow \infty} P(X_t = j | X_1 = j_1, \dots, X_L = j_L) = \pi_j.$$

The process defined by a MUCLICOP model is, according to Theorem 1, ergodic and has the same stationary distribution as the Markov chain defined by Q . Curiously, this distribution depends in no way on the λ 's. The proof of this result is very similar to that of Theorem 1 in Raftery(1985). It uses the standard trick for L^{th} -order Markov processes of forming the Markov chain $\{Z_t\} = \{X_t, \dots, X_{t-L+1}\}$. The stationary distribution of $\{Z_t\}$ is then manipulated to get Theorem 1. Note that if Z_L has the stationary distribution of $\{Z_t\}$ then $\{Z_t\}$ is stationary and, therefore, so is $\{X_t\}$.

Theorem 2.

Suppose that $\{X_t\}$ is stationary and let $P(n)$ be the $M \times M$ matrix with elements $p_{ij}(n) = P(X_{t+n} = i, X_t = j)$, $i, j \in \{0, \dots, M-1\}$, $n \in \mathbb{Z}$. (Thus $P(0) = \text{diag}(\pi_1, \dots, \pi_M)$.) Then

$$P(n) = \sum_{k=1}^K \gamma_k Q_k \sum_{l=1}^L \lambda_{kl} P(n-l).$$

The bivariate marginals of the stationary distribution of $\{X_t\}$ satisfy, according to Theorem 2, a simple matrix recursion. A proof of this theorem can be based on that of Theorem 2 in Raftery(1985).

3. Fitting MUCLICOP Models to Data by Maximum Likelihood

Let x_1, x_2, \dots, x_T be consecutive observations from an M -state process and $n(i_0, i_1, \dots, i_L) = n(i)$ be the number of

times that pattern i appears in the data stream, i.e.,

$$n(i_0, i_1, \dots, i_L) = \sum_{t=L+1}^T 1_{(x_t = i_0, x_{t-1} = i_1, \dots, x_{t-L} = i_L)}$$

The (conditional) log-likelihood of a K -component, MUCLICOP can be written

$$\begin{aligned} \log P(x_1, \dots, x_T | x_1, \dots, x_L) \\ &= \sum_{i: n(i) > 0} n(i_0, i_1, \dots, i_L) \log p(i_0 | i_1, \dots, i_L) \\ &= \sum_{i: n(i) > 0} n(i_0, i_1, \dots, i_L) \log \sum_{k=1}^K \gamma_k \sum_{l=1}^L \lambda_{kl} q_k(i_0 | i_1, \dots, i_L) \end{aligned}$$

The increased flexibility introduced by allowing than one model component makes enforcing nonnegative constraints on the λ 's in a MUCLICOP model much limiting than in an LCP model. Designing algorithms fitting MUCLICOP models with nonnegative λ 's is simpler than allowing them to take positive or negative values; therefore $\lambda_{kl} \geq 0$ for all k and l in the MUCLICOP model described in this paper. With these nonnegativity requirements, maximizing this log-likelihood with respect to γ 's, λ 's and Q 's is a linearly constrained optimization problem. Many techniques are available for problems like this. Fletcher(1987) and Gill *et al.*(1981) are excellent source optimization methods.

A 4-component, 20-lag MUCLICOP model for a state process has 439 independent parameters – many fewer than the 9×10^{20} for the general Markov model, but still many to use the generally preferred linearly constrained optimization method. (Newton's method would require solving a system of 439 equations at every iteration). MUCLICOP models of this size or larger are expected to be useful so an alternative to Newton's method is needed.

A nonlinear extension of the Gauss-Seidel algorithm known as *backfitting* is an optimization technique that has found many statistical applications. The idea behind backfitting is "divide and conquer." Arguments of the function whose optimum is desired are put in groups and the search for the optimizer proceeds one group at a time; this process is iterated. Almost any optimization technique can be used in backfitting. The backfitting algorithm that the authors have developed, BAFIM, uses the following parameter groups: $Q_1 = (q_1(i|j))_{M \times M}$, $\lambda_1 = (\lambda_{11}, \dots, \lambda_{1L})$, \dots , Q_K .

$\gamma = (\gamma_1, \dots, \gamma_K)$. BAFIM cycles through the parameter groups applying one linearly constrained Newton step per parameter group per cycle.

The backfitting strategy used in BAFIM greatly reduces the number of equations that must be solved at any one time; for the model in the above example, the number drops from 439 to a maximum of 90. Many models that would be too large to fit using Newton's method on anything but a super-computer can be fit on a workstation or even a personal computer using this backfitting algorithm.

Some of the parameters in a MUCLICOP model might have preassigned values; in particular, λ 's might have preassigned zero-values. The current version of BAFIM can handle λ 's with preassigned zero-values but BAFIM can generate zero-valued parameters as well. BAFIM uses an active set strategy to handle parameter values on the boundary (see Chapter 5 of Gill *et al.* (1981)).

As with other iterative techniques, different initializations of the BAFIM algorithm can result in different fitted models with similar likelihoods. MUCLICOP models form a non-identifiable class; it is unsurprising that many different-looking models might fit the data about equally. A good start can make the difference between an interpretable model and one that fits well but seems nonsensical. Care must be taken in finding these good starts. The authors have more experience fitting MUCLICOP models to binary data than to data with $M > 2$; initialization in the binary case is focused on here.

The authors have found that the sample autocorrelation function (ACF) gives good indications of the order L required to fit the data with a MUCLICOP model. The ACF can be used to pick L by setting it equal to the largest lag at which a significant autocorrelation is present. (For "significant" here, in the above discussion of the spectrum and the remainder of this paper, read "not small.") The sample ACF can also be used to pick K , the number of components. Recall that the λ 's in a MUCLICOP model are constrained to be nonnegative. A single component (one Q) cannot, therefore, favor a 1 now given a 0 at one lag and a 0 now given a 0 at a different lag; positive and negative autocorrelations imply the need for a multicomponent model. In the initial stages of model fitting, the authors believe it is better to include too many components in the model than too few. If correlations of the same sign occur at sets of lags that are separated, one should fit a

model with a component for each set. It may turn out that a single component is all that is necessary but it is better to allow for the possibility of two different generating mechanisms at the two sets of lags. The basic idea, then, is to group neighboring lags with similar significant autocorrelations together and assume that each group requires a component.

Once the number of components K has been identified the initialization of the parameters themselves can be addressed. In the absence of *a priori* information, it is unreasonable to initialize each entry of γ with $1/K$. The process of identifying the number of components based on ACF described above produces groups of lags. Information related to these groups can be used to initialize the Q 's and λ 's. Experience has shown that the initialization of the Q is more critical. Let n_k denote a representative lag near the middle of the k^{th} group and let

$$\hat{p}_{ij}(n) = \frac{1}{T - n + 1} \sum I_{(x_{i+n} = i, x_i = j)}.$$

By taking the starting values for Q_k as

$$Q_k = \left(\hat{p}_{ij}(n_k) / \sum_{i=0}^{M-1} \hat{p}_{ii}(n_k) \right),$$

one is using estimates of the conditional probability $P(X_{i+n_k} = i | X_i = j)$, $i, j \in \{0, 1 (=M-1)\}$ to initialize the Q -matrices. (These initial estimates tend to be well in the parameter space whereas the fitted values tend to be on the boundary.) Finally, there are a number of ways to initialize λ 's. One method that has proved useful is to set λ_k to the normalized absolute values of the autocorrelations for the lag in the k^{th} group. (Sometimes the authors have found it useful to set the λ_k entries for lags outside the k^{th} group to a small positive value prior to normalizing so that BAFIM starts with estimates in the interior of the parameter space.)

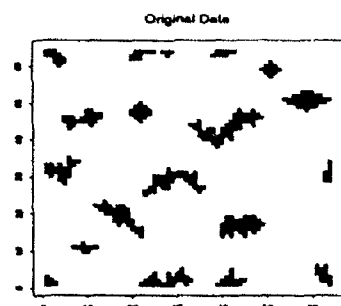
As an alternative method of initializing the parameters of a MUCLICOP model, one could turn Theorem 2 around, fixing the P 's at their sample estimates $\hat{P}(n) = (\hat{p}_{ij}(n))$ and solving, for various n -values,

$$\hat{P}(n) = \sum_{k=1}^K \gamma_k Q_k \sum_{l=1}^L \lambda_{kl} \hat{P}(n-l)$$

for γ , the Q 's and the λ 's. This alternative method would work for any size state space. Unfortunately it is also a nonlinear problem.

4. An Example

The data for the example is a 64×64 raster plot of 0's (white) and 1's (black). Since the data are written on a torus,



one should see significant lags at multiples of 64. Based on the ACF of the data, the authors constructed a two component model each component involving 200 lags.

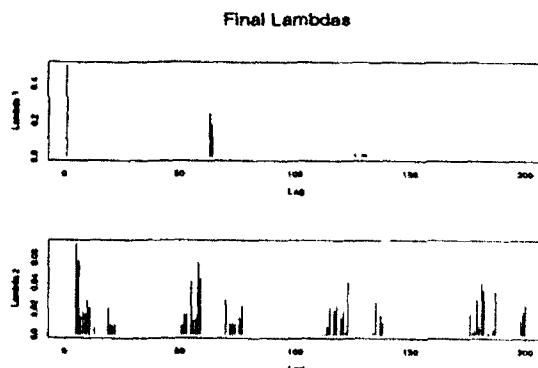
The parameters of the model were estimated as $\hat{\gamma}_1 = 0.99$, $\hat{\gamma}_2 = 0.01$,

$$Q_1 = \begin{array}{cc} & \begin{matrix} 0 & 1 \end{matrix} \\ \begin{matrix} 0 \\ 1 \end{matrix} & \begin{pmatrix} 1 & 0.02 \\ 0 & 0.98 \end{pmatrix} \end{array}$$

and

$$Q_2 = \begin{array}{cc} & \begin{matrix} 0 & 1 \end{matrix} \\ \begin{matrix} 0 \\ 1 \end{matrix} & \begin{pmatrix} 0.43 & 1 \\ 0.57 & 0 \end{pmatrix} \end{array}$$

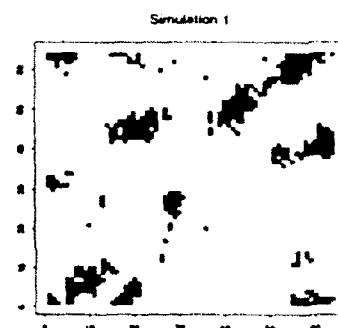
The λ 's display the expected periodicity at the appropriate



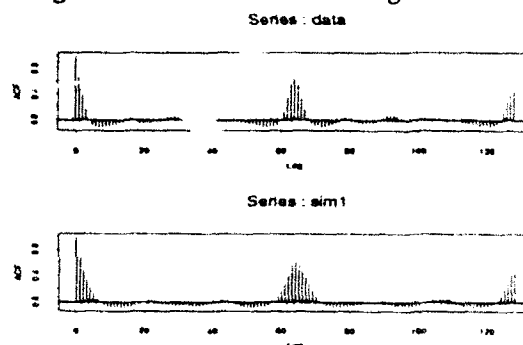
lags. (Note that the above plots are on differing scales.)

This fit is easy to interpret in light of the actual data: majority of the time the process uses the first component which stays in current state yielding runs of 0's and 1's; second component forces a 1 to transit to a 0.

A simulation based on the this model shows that



the model generates "islands," the runs of 1's are longer in the original data. The ACF of the original data vs. the



the simulated data reinforces this impression. Perhaps the effect could be reduced if $\hat{\gamma}_2$ were set to a larger value, say.

5. Concluding Remarks

The backfitting algorithm introduced in Section 3 is of course, the only alternative to Newton's method for generating (local) MLE's of a MUCLICOP model. Gradient search methods are simple alternatives that do not require equating and do not need to calculate or store approximations to the Hessian. They do not depend on the objective function being similar to a quadratic function near the optimum for their convergence properties. Gradient search methods are however, slow. An EM algorithm similar to those of McLerran (1990) and Schimert (1990) for mixture transition distribution models could possibly be developed for fitting MUCLICOP models.

COP models. Such an approach might prove useful if the X -process can only be observed with "noise."

Many other parsimonious model families have been proposed for high-order, finite state, Markov processes. In a series of papers (Jacobs and Lewis(1979a, b, c; 1983)), Jacobs and Lewis developed a family that's analogous to ARMA models for gaussian time series. Also see Logan(1981). MUCLICOP models are, in the authors' opinion, simpler to use and interpret in practice than are these alternatives.

References

- Fletcher, R. (1987). *Practical Methods of Optimization*, 2nd Edition. John Wiley & Sons, New York.
- Gill, P.E., Murray, W. and Wright, M.H. (1981). *Practical Optimization*. Academic Press, New York.
- Jacobs, P.A. and Lewis, P.A.W. (1978a). Discrete time series generated by mixtures I: Correlational and runs properties. *JRSSB*, 40, 94-105.
- Jacobs, P.A. and Lewis, P.A.W. (1978b). Discrete time series generated by mixtures II: Asymptotic properties. *JRSSB*, 40, 222-228.
- Jacobs, P.A. and Lewis, P.A.W. (1978c). Discrete time series generated by mixtures III: Autoregressive processes ($DAR(p)$). Naval Postgraduate School Tech. Rep. NPS 55-78-022.
- Jacobs, P.A. and Lewis, P.A.W. (1983). Stationary discrete autoregressive-moving average time series generated by mixtures. *J. Time Series Anal.*, 4, 18-36.
- Le, N.D., Martin, R.D. and Raftery, A.E. (1990). Modeling outliers, bursts and flat stretches in time series using mixture transition distribution models. University of Washington Tech. Rep. 194.
- Logan, J.A. (1981). A structural model of the higher-order Markov process incorporating reversion effects. *J. Math. Sociol.*, 8, 75-89.
- Raftery, A.E. (1985). A model for high-order Markov chains. *JRSSB*, 47, 528-539.
- Schimert, J. (1990). Parsimonious higher order hidden Markov models. Unpublished University of Washington Tech. Rep.
- Tavaré, S. and Raftery, A.E. (1990). Estimation in the linear conditional probability model for high order Markov chains. University of Washington Tech. Rep. 211

Stochastic Epidemics on a Grid: Endemicity and Related Problems

Michael Lloyd

Department of Actuarial Mathematics and Statistics
Heriot-Watt University
Riccarton
Edinburgh EH14 4AS
Scotland

Abstract

Lattice-based nearest-neighbour contact models have been proposed in the past as useful models for the spatial aspects of infectious disease behaviour. Such models are linked to established areas of research such as percolation theory and cellular automata. Despite these connections, progress towards a general understanding of their dynamic properties has been slow. This paper presents results from recent work involving extensive computer simulation of a particular epidemic model. Some discussion is made of the issues in visualization in this context, and some graphics are used to illustrate mathematical results for the underlying stochastic process.

1. Introduction

The model considered here comes from the general class referred to as Interacting Particle Systems which have been receiving increased attention in recent years, at least in part due to the increasing capacity of computers which aid in the visualization of, and in the formation of hypotheses about, such systems. This particular model has been hypothesized for use in epidemic modelling by Mollison and Kuulasmaa [5] (where it was used in the context of rabies transmission in European fox populations). It is also related to the forest fire models considered in the central work of Cox and Durrett [1] (and references therein), and particularly closely to the recent advance in Durrett and Neuhauser [2]. These models are all straightforward to describe, driven by a small number of possible interactions between an individual and a specified set of neighbours. However, they have proved to be anything but trivial to analyze. After several decades of effort, rigorous results are still confined to asymptotics of various sorts. Very little is known about the finite-time dynamic behaviour of these systems, or about their approach to the known limiting distributions or states, although a number of hypotheses have been aired.

In the proceedings of the previous Interface confer-

ence, work was presented in Lloyd [4] which took an empirical approach to some questions concerning a simpler model. Repeated simulation runs were collected and hypotheses drawn; these hypotheses were more thoroughly tested in [3]. This paper draws on a similar philosophy of empirical testing to elucidate areas where formal mathematical results are not available. Specifically the topic of endemicity, or long-term stability of a trivial state, is considered for a model containing basic processes of recurrent epidemics: infection, death and rebirth. It is stressed that this model is not tied to its epidemic interpretation, nor is it posited as a realistic description of any particular disease. Rather, it is a core model type which must be fully understood before more complex infrastructures are built upon it in application of spatial spread modelling.

2. The Model To Be Considered

Divide the plane \mathbb{R}^2 into a regular array of hexagons; for definiteness, arrange that the centre-to-centre distances between neighbouring hexagons is 1, that the hexagon centred on the origin, and that each hexagon is oriented with two sides parallel to the x axis. Each hexagon is thought of as representing an individual; will initially be assigned to one of the three states in the model: *susceptible*, *infectious* and *empty*. We specify three transitions, as follows:

Infection: at rate 1, an infectious individual randomly selects one of its nearest neighbours; if susceptible this site becomes infectious.

Death: at rate δ , an infectious site becomes empty.

Rebirth: at rate ρ , a susceptible individual randomly selects one of its nearest neighbours; if empty, this site becomes susceptible.

An event which happens 'at rate r ' occurs at the event time of a Poisson process of intensity r . All processes specified above are independent.

Having established this process, we can readily imagine an outline of what will start happening if we begin the process with all sites susceptible apart from the origin, which is to be infectious. So long as the rate δ is slow enough¹, we would expect an epidemic to get going with high probability. Such an epidemic would take the form of an expanding cluster around the origin. As time passes, this area will continue to expand, but the first infected sites (near the centre of the cluster) will begin to become empty. We might guess that a ring shape will emerge, with a wave of infection spreading into the plane of susceptibles, but leaving behind an empty area². If any susceptibles manage to get 'inside' this ring, then they will be able to regrow into a fresh susceptible population, and it is here that the real complications begin. There is no longer a ready intuition at this point to say what the model will do.

Computer animation and visualization techniques can be of immense assistance at this point, allowing many experiments to be performed and observed, with a view to formulation of hypotheses about the range of behaviour of the above system. Clearly, the above regular tessellation of space, the small number of states, and the simple (independent) transition rules mean that the model lends itself well to computer simulation (over a finite area, with periodic boundary conditions). It is perhaps slightly unusual to prefer the hexagonal lattice structure (a square lattice being more common in the literature); see Lloyd [3] for a discussion of the details in implementing this form, and also for a demonstration of its superiority over four neighbour and eight neighbour square forms for a particular (related) growth model.

3. Empirical Approach

The first feature of the model considered here is endemicity. In the real world, this term refers to a disease which can sustain itself indefinitely in a given population without the need for introduction of fresh infectious individuals. When simulating the above process over a finite section of the lattice, however, it is clear that true endemicity is impossible; with only a finite number of possible configurations of the system and the fact that the model is ergodic, we know we shall always reach an absorbing state with no infectious individuals in a finite amount of time. However, it is conceivable that the model on the full infinite lattice does not have this problem; indeed, Durrett and Neuhauser [2] have proven that non-trivial stationary distributions exist for a very

similar model, and have hypothesized that they also exist for the model defined here. It is then reasonable to expect that model settings which will persist indefinitely on an infinite lattice will usually last for a 'large' amount of time on a moderate-sized finite area, and conversely, those which die out on the infinite lattice will do so quickly and are likely to behave in the same way on a finite section. The following experimental scheme is therefore suggested:

1. Select a cut-off time T to represent 'endemicity'
2. Pick δ and ρ at random from some reasonable range of values
3. Simulate the (δ, ρ) model for up to a maximum of T time units
4. Record t , the time to extinction of the epidemic or the value T if the model still contains infectious sites
5. Repeat 2-4 a large number of times to cover the range of possible (δ, ρ) values

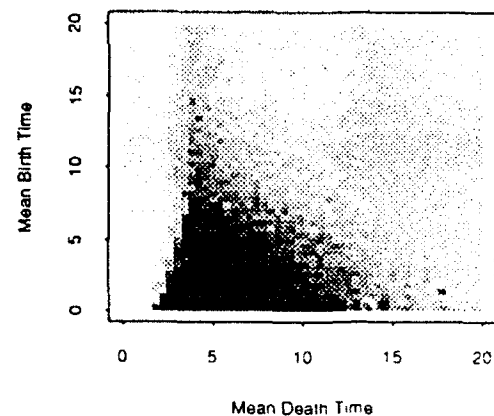


Figure 1: *Local Means for Time to Extinction of the Epidemic Model*

This scheme has been implemented, and the output is shown in Figure 1. Note that interesting values of δ and ρ would be very small since the infection rate of δ must be relatively large for an epidemic to get going, so the parameters are replaced in this plot with the mean time to next event in the respective processes, i.e. $\frac{1}{\delta}$ and $\frac{1}{\rho}$. The range chosen for the mean times was 0 to 20 for each parameter, and parameter pairs were selected using a uniform distribution over this rectangle. Each square in the plot is shaded according to the mean time to absorption of all the simulations whose parameter

¹'Slow enough' would formally mean something corresponding to standard threshold results - that if δ is less than some critical value δ_c there will be an epidemic with non-zero probability.

²akin to the shape theorem of Cox and Durrett [1]

fell within that square, with paler shadings representing shorter times and black representing $t = T$, models which were still not absorbed at the cutoff time, here set at 1000.

4. Visualization Issues

There is a great deal of interesting information to be extracted from the observations underlying Figure 1, but the difficulties in presenting all the available detail are substantial. In essence, the data are naturally in the form of a scattercloud: a set of (x, y) points $(\frac{1}{\delta}, \frac{1}{\rho})$, each with a single z value t . However, conventional rotating cloud techniques do not give a particularly strong sense of the nature of the data, are also rather slow when dealing with many tens of thousands of observations, and are notoriously awkward for publication purposes. The first problem relates to the fact that the parameter space does not divide up cleanly into phase regions in the conventional sense; that is, wherever endemicity occurs, it occurs only with a certain probability. Observations in such a region of parameter space will also produce an appreciable number of failed epidemics, so the t observations in this area will have a distribution rather than a single value (albeit a distribution concentrated largely on the value T). The human eye is not especially good at making deductions about such distributions when presented with the information in the form of a rotating cloud, where the patterns are shifting throughout the plot and the density of points is very high.

As an alternative, various fixed plots have been used in the analysis of this data, most notably localized means plotted in grey scales, like Figure 1. However, this plot did require some manipulation before the desired features were clear; it does not, in fact, show simple means in each region. Such a plot suffers from the phenomenon of failed epidemics mentioned above, resulting in the black 'endemic' region being harder to see. To overcome this, all observations below 75 were removed before local means were calculated. This seemingly arbitrary alteration to the data will be justified below.

A number of other approaches have been used to portray the features of this dataset, but space restrictions preclude their reproduction here. Successful alternatives and complements to the above include three-dimensional surface representations of the local mean data (both treated and untreated by the lower cutoff of 75), various contouring techniques (which were useful for comparative overlays of related datasets), and two-dimensional scatterplots of the unaggregated data where colour coding over some simple scale (typically 2 or 3 colours) was used to represent the third variable t .

5. Breakdown of the Phase Portrait

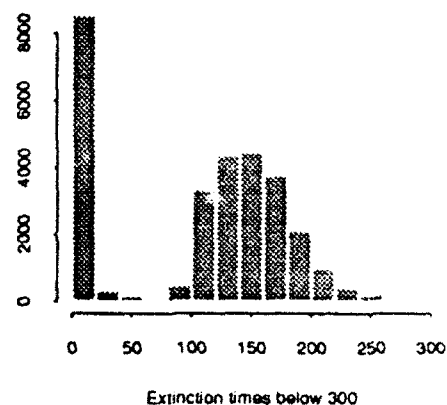


Figure 2: *Distribution of Extinction Times up to 300*

To explain the arbitrary alteration to the data of Figure 1, consider Figure 2. This plot shows a histogram of the observed t values which lie below 300; what is not shown (so that detail can be resolved) is the near zero density to the right of the plot until 1000 (= T). This distribution is therefore trimodal, with two readily explainable peaks: many observations at the endemic cut-off T (not shown) or at values very close to 0, corresponding to endemicity or near-immediate failure of epidemic respectively. However, there is a third region apparently centred around times of 140. Using techniques such as plot linking, it is possible to investigate the spatial organization of these three modes in phase space, and we very quickly observe Figure 3.

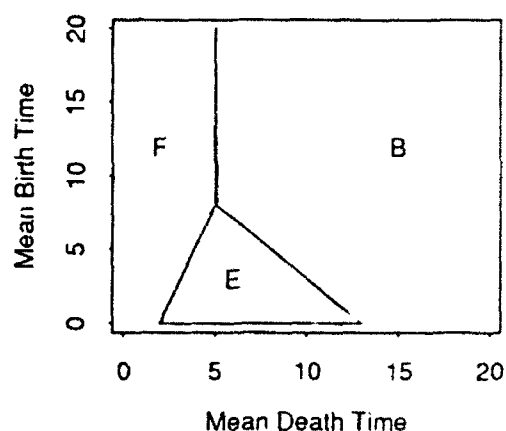


Figure 3: *Schematic Breakdown of Phase Space Diagram, showing Failure, Endemicity and Burn-out regions*

This diagram represents the three observed regions of phase space in this model, notwithstanding the po-

made earlier that these regions are less sharply defined than in other fields where phase diagram regions are delineated. Instead of saying that a particular event will happen in a region, we sometimes can say only that there is a non-zero probability of that event in the region and that it does not occur outside.

The Failure region consists entirely of points near zero in Figure 2; this situation is normally referred to as an epidemic 'below threshold', or with $R_0 < 1$, meaning that the expected number of new infections produced by a single site is less than one and hence by analogy with branching processes, any cluster of infection must die out at least exponentially fast. The Endemic region contains those parameter pairs for which there will be a long-lasting epidemic so long as there is no early failure to spread. Consider again the outline of behaviour in Section 2; this endemicity will generally take the form of an expanding ring which leaves behind a nearly empty region, but the few susceptibles which are passed over repopulate the lattice, and eventually this new population will encounter an infectious individual left from the original wave, which will start the process again. However, if the regrowth process is too slow or too few susceptibles are left by the first wave, we can imagine situations where the new population does not encounter any infectious sites before it becomes dense enough to support a new wave. This gives the phenomenon of Burnout, where the infection is too devastating in its effect to be sustained in a finite population. The expanding ring described in Section 2 moves out to the edge of the simulation area and has to die out, because there are insufficient numbers of susceptibles for it to continue. This explains the third concentration in Figure 2; the time for a burnout to occur is roughly fixed since the infection rate is set at 1, so the time to reach the edge of the simulation area is always the same (the small variation then comes from the time to death of the last infectious site, which depends only on δ).

Now to see these regions clearly in the local mean plot of Figure 1, we note the density of the 'troublesome' failure events: it is 1 throughout the Failure region, and then decreases to the right but does not move to 0 very quickly. Hence to observe the important distinction between the Endemic and Burnout regions we remove all the early failures, using the cut-off of 75 suggested by Figure 3.

6. The Nature of Endemicity

Having observed the Endemic region, we can now begin to ask questions about what this endemic behaviour is; certainly the intuitive description in Section 2 does little to suggest what any 'equilibrium' would be like.

Again, the development of ideas to do with this question is greatly helped by animated simulation. With the aid of the above phase space portrait, custom simulation software can be set to display various parts of the Endemic region, looking for visual characterizations and variations.

The most important conclusion to come from such a visual analysis is the oscillatory nature of the model. That is, there is no apparent convergence to a stable equilibrium with a certain prevalence of infection. Instead there is large-scale organization into complex clusters of similar states, and from this comes a form of 'boom and bust' quasi-periodic behaviour in the number of infectious sites present. To illustrate this, Figure 4 shows the oscillations in prevalence for one particular model. These oscillations are remarkably stable and pervasive; they appear to occur throughout the endemic region, and, if disturbed (by, say, randomly 'stirring' all the sites) they quickly re-assert themselves.

A full explanation of these oscillations is proving elusive, but they certainly point to very rich behaviour in the model. Clearly, it is necessary to be very careful about what is meant by an 'equilibrium' state here; asymptotic results (where they can be found) will need to be sufficiently sophisticated to incorporate the fact that for all finite times, it appears that local prevalence levels do not stabilize. The oscillations are also not particularly easy to characterize; visually, they appear to have fairly regular amplitude, and the upward and downward slopes appear quite regular from peak to peak. However, the spacings of the peaks are not regular -- there appear to be 'pauses' of random length between successive cycles -- and this renders most usual frequency domain techniques invalid. A straightforward Fourier transform of this series has a surprisingly broad peak due to this irregularity. Work is continuing on this point.

7. Finite Size Effects

It is worth making a few points about finite size effects here. It is certain that the burnout phenomenon described above is dependant in its actual occurrence on the grid size; it will take longer and longer for the expanding ring to move through the full population as the grid size increases. Therefore we can hypothesize that the boundary between the Endemic and Burnout regions in Figure 3 will move as grid size changes. We expect that larger grids will give more possibilities for subsequent waves to get started. However, it is unclear whether the region disappears altogether over the infinite lattice. The hypothesis of Durrett and Neuhauser [mentioned in Section 3] would imply that it does disappear, but this has yet to be rigorously proven. Whether

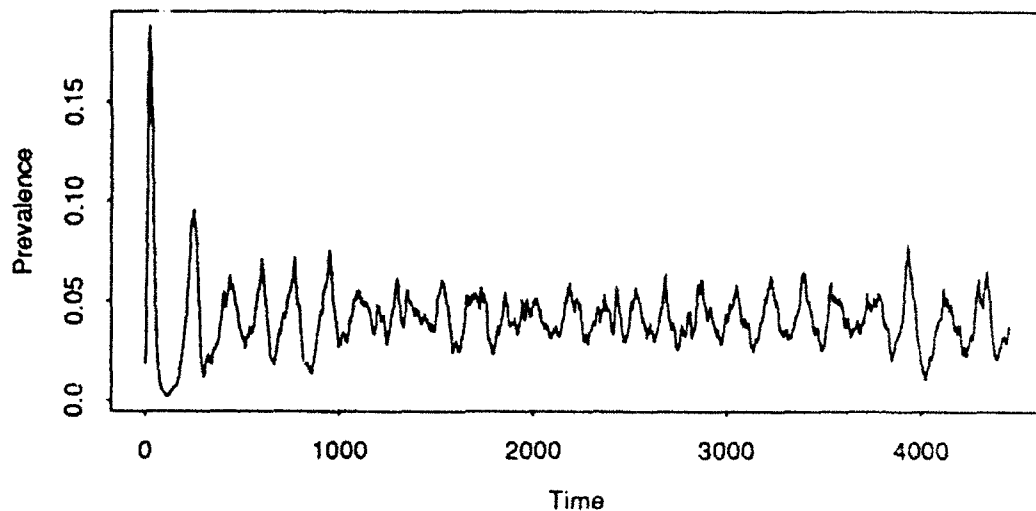


Figure 4: *Proportion of Infectious Sites over Time with $(\delta, \rho) = (1, .1)$*

it is proven or not, it is stressed that this does not make the phenomenon less interesting; the fact that it only occurs in finite populations is of much relevance when we recall that in almost any application of such modelling, the population under consideration will be finite. The infinite lattice case may make analytical work easier, but it appears that it may miss out on a highly significant effect. As an example, consider a disease moving through a spatially organized population (such as rabies amongst foxes [3]). The infinite lattice model would predict that any sufficiently virulent disease will be sustained forever, whilst a model which incorporates the finiteness of the real fox population points to diseases which are far too destructive to be sustainable, implying an upper threshold phenomenon in addition to the usual lower one.

Similarly, the oscillations in prevalence can be attacked as being only a finite size effect, but this too is not necessarily meaningful. Certainly, on an infinite lattice where an infinite number of such regions are oscillating out of phase, there will be stable prevalence, but any localized observation will show patterns such as Figure 4. Again, in real world applications this is exactly what would be done. When considering the effect of a disease process, it is usual to be interested in only a small area (a state or county) which is a subsection to a much larger process (which we may wish to model as infinite). Hence these oscillations are also a finite size effect, but none the less interesting for that.

Acknowledgements

I would like to thank my Ph.D. supervisor, Prof. Denis Mollison, and both Professors Richard Durrett and David Griffeath with whom I had some most helpful discussions about the work presented here.

References

- [1] J.T. Cox and Richard Durrett. Limit theorems the spread of epidemics and forest fires. *Stoch. F. Appl.*, 30(2):171-191, 1988.
- [2] Richard Durrett and Claudia Neuhauser. Epidemics with recovery in $d=2$. *Ann. Appl. Probab.*, 1(2):1206, 1991.
- [3] Michael Lloyd. The effect of grid shape on an epidemic growth model. Unpublished manuscript.
- [4] Michael Lloyd. Some results in the simulation analysis of the shape of spread of epidemics on a grid. In *Proc. 23rd Symp. on the Interface*, pages 487-499, 1991.
- [5] Denis Mollison and Kari Kuulasmaa. Spatial epidemic models; theory and simulations. In P.J. Condon, editor, *The Population Dynamics of Rabies Wildlife*, pages 291-309. Academic Press, London, 1985.

On a Family of Autoregressive Processes and Cantor Sets

Chamont Wang, Trenton State College
and
Arthur Silverberg, American Cyanamid

Abstract

We will present a variety of fractal images derived from a random chaotic game. We also prove certain mathematical properties of a family of Cantor sets, the compact supports of certain autoregressive processes. In addition, we investigate the distributions, "densities", Poincare maps, the fractal dimensions and the Renyi information dimensions of the time signals.

TIME SERIES AND POINCARÉ MAPS

Consider a univariate time series such as the one in Figure 1 (sample size $n=1000$):

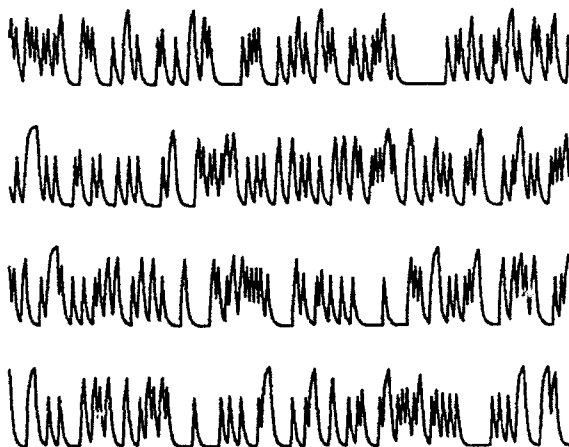


Figure 1

The problem here is to discern patterns in the series, or hopefully to uncover the mechanism that generated the data. Given the above time signals, eyeball examinations by 42 students in classes at Trenton State College disagreed on repeated patterns or quasi-periodicity in the graph. A histogram of the data (using 200 cells) looks like the New York skyline, see Figure 2.

The graph shows a distinct pattern which is not like that of a Gaussian or any other commonly used distribution. A scatterplot of X_{n+1} versus X_n is shown in Figure 3.

The plot reveals two Cantor sets with identical slopes. The equations of the two lines are

$$\begin{aligned} X_{n+1} &= .4X_n + .4 \quad \text{and} \\ X_{n+1} &= .4X_n. \end{aligned}$$

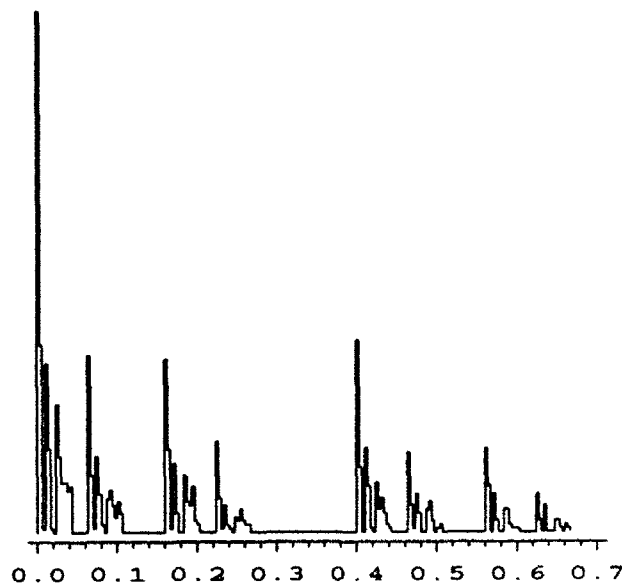


Figure 2

Using these two equations while noting that about 70% of the data are on the left of the histogram, we infer that the mechanism which generated the graph in Figure 3 (and hence the graph in Figure 1) is given by the formula:

$$X_{n+1} = \lambda I_{n+1} + \lambda X_n, \quad (1)$$

$$I_{n+1} = \begin{cases} 1 & \text{with probability } p, \\ 0 & \text{with probability } 1 - p, \end{cases}$$

where $\lambda = .4$ and $p = 0.3$. In the statistical time-series literature (see, e.g., Box and Jenkins, 1976), tests for the goodness-of-fit of the model include primarily residual analysis, R^2 , χ^2 statistics, etc. In this example, we will take a different route to assess the validity of the model in (1).

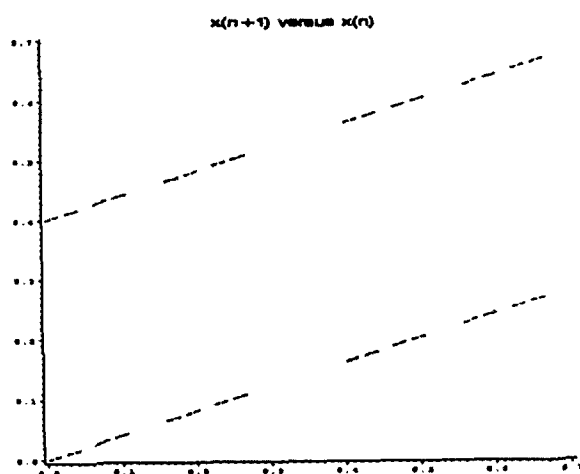
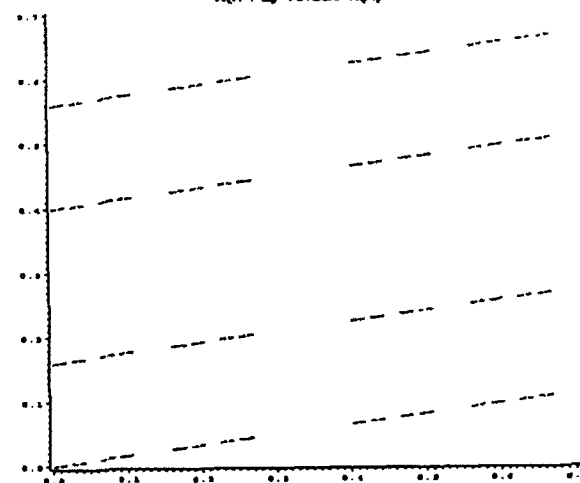
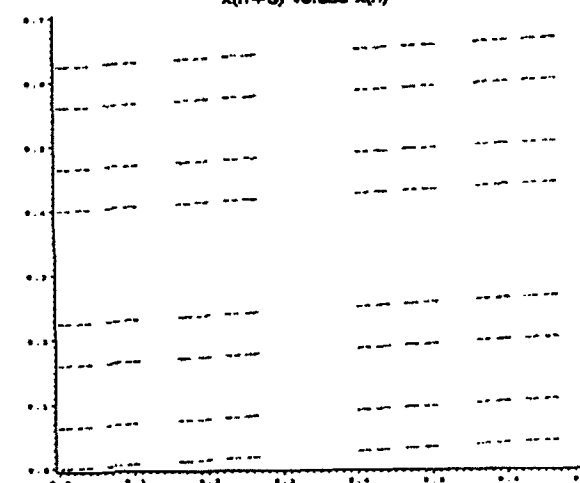
Figure 3
 $x(n+2)$ versus $x(n)$ Figure 4
 $x(n+3)$ versus $x(n)$ 

Figure 5

We first plot the scatter diagrams for X_{n+L} versus X_n , L and 3^1 , (see Figures 4 and 5). These diagrams are called Poincare maps in the chaos literature.

These graphs lend support to the model in (1). For instance, by (1) we have

$$X_{n+2} = .4I_{n+2} + .16I_{n+1} + .16X_n$$

Note that there are four different possibilities of (I_{n+2}, I_n) and indeed there are four different line segments with slope .16. The same argument applies to the case $L = 3$.

There are other pieces of evidence that can be used to support the model in (1). First, AR(1) model in (1) is a stationary process; and apparently the graph in Figure 1 supports this conclusion. Second, $\text{Cov}(X_n, X_{n+L}) = (.4)^L$, which decreases as L increases. This is another fact that matches the Poincare maps given in Figures 3, 4 and 5. The sample correlation coefficients ($n = 1000$) are 0.422, 0.162 and 0.063 for lag 1, 2 and 3, respectively. Another major question in the study concerns the missing parts in these Cantor lines. By (1),

$$X_n = \lambda I_n + \lambda^2 I_{n-1} + \dots + \lambda^n I_1 + \lambda^n X_0,$$

where $\lambda = .4$. Hence

$$X_n \begin{cases} \leq \frac{\lambda}{1-\lambda} & \text{for all } n; \\ \leq \frac{\lambda^2}{1-\lambda} & \text{if } I_n = 0; \\ \geq \lambda & \text{if } I_n = 1. \end{cases}$$

For $0 \leq \lambda \leq 0.5$, it is therefore impossible for X_n to be between $\lambda^2/(1-\lambda)$ and λ . In general, if $\lambda I_n = \dots = \lambda^k I_{n-k+1} = 0$ (i.e., X_n is in the far left segment), then

$$X_n \begin{cases} \leq \frac{\lambda^{(k+1)}}{1-\lambda} \\ \leq \frac{\lambda^{(k+2)}}{1-\lambda} & \text{if } I_{n-k} = 0, \\ \geq \lambda^{(k+1)} & \text{if } I_{n-k} = 1. \end{cases}$$

It is therefore impossible to have an X_n between $\lambda^{k+2}/(1-\lambda)$ and λ^{k+1} . Formula (4) also implies that the length of the right part equals the length of the left part $= \lambda^{(k+2)}/(1-\lambda)$, that the length of the middle (missing) part is $\lambda^{(k+1)}(1-2\lambda)/(1-\lambda)$. The total length of the missing part is $\lambda/(1-\lambda)$. Therefore the total length of the Cantor set equals zero.

It can be shown that the distribution functions $F_n(t) = \text{Prob}[X_n \leq t]$ obey the following iterative relationship:

$$F_n(t) = pF_{n-1}\left(\frac{t}{\lambda}\right) + (1-p)F_{n-1}\left(\frac{t}{\lambda} - 1\right)$$

¹ Poincare delay maps with lag=1 and 2 were shown to students. The following question was then asked: "How many Cantor lines are we going to have if lag=3?" Sixty percents of the students said "six lines," the remaining forty percents said, "8 lines".

Let $u_n = E[X_n]$, then by algebraic manipulations one can show that $u_{n+1} = \lambda u_n + \lambda p$. When $p = 0.5$ we can also show that

$$F_n(u_n - t) = 1 - F_n(u_n + t) \quad (6)$$

Therefore, F_n is symmetric about the mean, u_n , when $p = 0.5$.

In (1) we may also investigate the time series if λ is negative. If $-1 < \lambda < 0$ (e.g., $\lambda = -.4$), then the Poincare maps remain the same for even lags but have negative slopes for odd lags. To investigate the support for negative λ we may assume $p = 0.5$ since the support does not depend upon p . F_n is symmetric about the mean when $p = 0.5$ and λ is negative. The odd moments are therefore zero and the even moments are functions of λ^2 . Therefore since the support is a finite interval, the distribution functions for λ and $-\lambda$ are of the same type (see Feller, 1971). We have therefore proven that the ratio of the major middle missing part to the whole length is $1 + 2\lambda$ (0.2 if $\lambda = -.4$) as is the case for positive λ .

The Kolmogorov capacity of the compact support of X_n in (1) is

$$D = -\frac{\ln(2)}{\ln(\lambda)} \quad (7)$$

Note that $D = 0.756$ or 0.431 if $\lambda = 0.4$ or 0.2 , respectively. By comparison, the Lebesgue-measure of the Cantor sets equals 0, no matter whether $\lambda = 0.4$ or $\lambda = 0.2$. The related Reyni information dimension is

$$D_I = \frac{p \ln(p) + (1-p) \ln(1-p)}{\ln(\lambda)} \quad (8)$$

If $\lambda = 0.4$ and $p = 0.5$, then $D_I = D = 0.756$, but if $\lambda = 0.4$ and $p = 0.3$, $D_I = 0.667$.

DENSITY FUNCTIONS WITH INFINITELY-MANY SINGULARITIES

Given the first-order autoregressive model

$$X_{n+1} = \lambda \times (X_n + I_n), \quad (9)$$

with $p = 0.5$, this section investigates the "probability density function" of X_n as n approaches infinity. We put "probably density function" in quotes since for singular distributions the density function is zero almost everywhere. For $\lambda = .6$, simulating $n = 10,000$ iterations (after ignoring the initial 100 iterations) the histogram of X_n is shown in Figure 6.

The graph appears self-similar and seems to have infinitely many singularities when the density function $f(t) \sim 0$. We note that the histogram for X_n when $\lambda = .5$ is a rectangle and it can be proved it gives a uniform distribution. Also of interest is the histogram when $\lambda = 0.7$, Figure 7.

At this moment, we do not have a mathematical explanation of the phenomena when $\lambda > .5$. A conjecture of the

singularities (when λ is near 0.6 and $f(t) \sim 0$) is:

$$\text{the singularities occur at } \frac{\lambda^{(k+2)}}{1-\lambda^2} \text{ and} \quad (1) \\ \text{symmetrically at } \frac{\lambda}{1-\lambda} - \frac{\lambda^{(k+2)}}{1-\lambda^2}.$$

Assume that X_0 is a random number from the unit interval so $F_0(t) = t$, $0 < t < 1$. By (5), the precise functional form of $F_n(t)$ and hence the probability densities $f_n(t)$ can be calculated accordingly. For instance when $0.5 < \lambda < p = 0.5$,

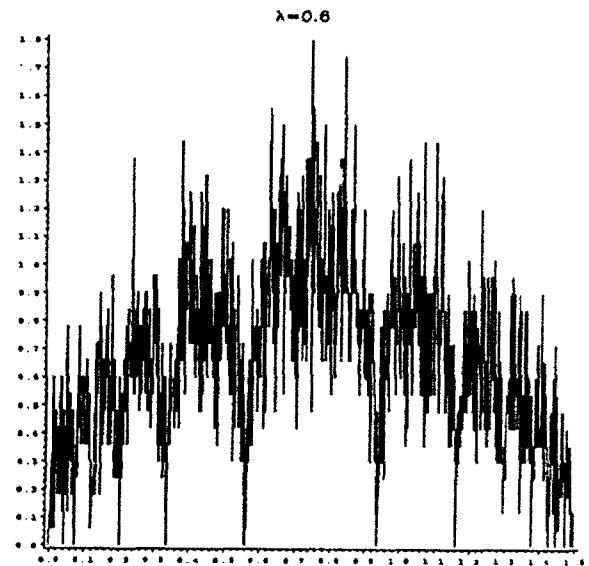


Figure 6
 $\lambda = 0.6$

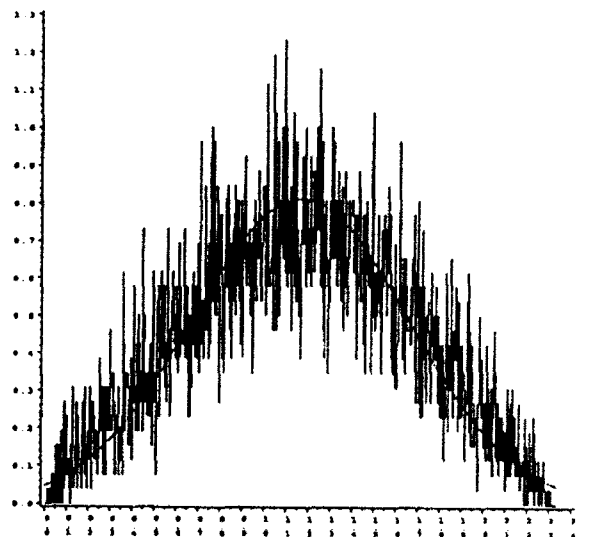


Figure 7

$$\begin{aligned}
 F_1(t) &= t/(2\lambda) & \text{for } 0 \leq t \leq 2\lambda \\
 F_2(t) &= t/(4\lambda^2) & \text{for } 0 \leq t \leq \lambda \\
 &= t/(2\lambda^2) - 1/4\lambda & \text{for } \lambda \leq t \leq 2\lambda^2 \\
 &= .5 + t/(4\lambda^2) - 1/4\lambda & \text{for } 2\lambda^2 \leq t \leq \lambda + 2\lambda^2
 \end{aligned} \quad (11)$$

Here are graphic representations of the density functions f_n for $n = 1, 2, 3, 4, 5$ and 10 when $\lambda = 0.6$ (see Figures 8-13).

The graphs and calculations similar to those of (11) lead to the next conjecture.

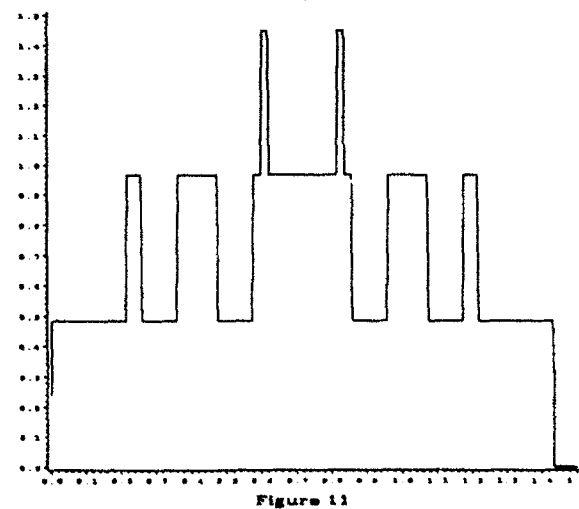
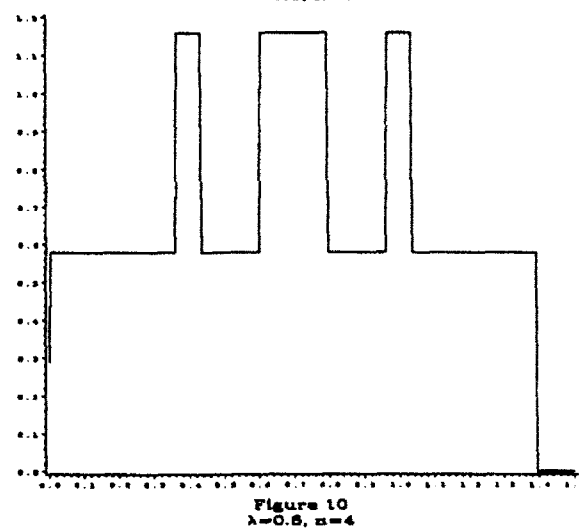
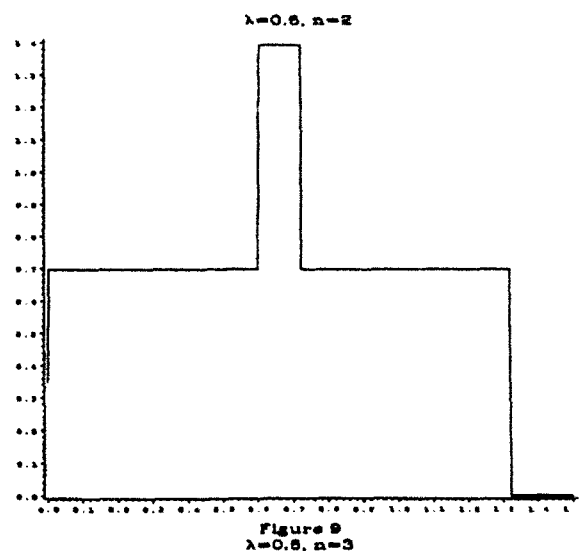
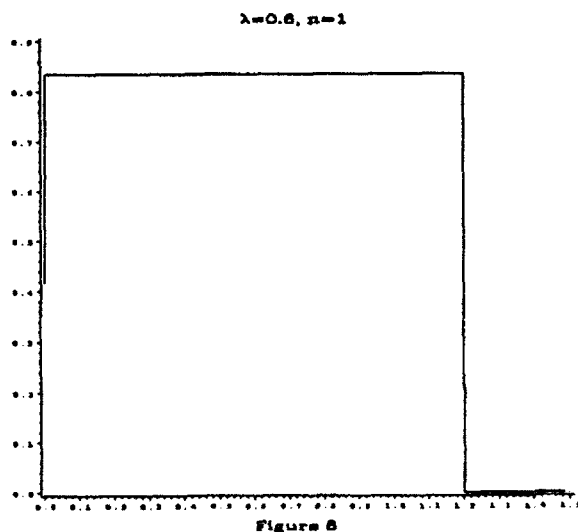
The derivative of the density function
(as $n \rightarrow \infty$ and λ is in a neighborhood of 0.6) (12)
does not exist a.e.

Note that if $\lambda = .7$ (Figure 7), the histogram (simulation $n = 10,000$ iterations) is close to a bell-shaped curve. For this reason, we further investigate the cases where $\lambda = 0.617, 0.618, 0.61806$ and 0.619 (see Figures 14-17), near the region of the conjectured singular point at $\lambda^2/(1 - \lambda^2)$. These four Figures were calculated by a numerical approximation not a simulation.

At $\lambda = 0.617$ and 0.618 , the curves almost touch the x-axis at $x = \lambda^2/(1 - \lambda^2) \sim 0.6147$ and 0.6179 , respectively. The case of $\lambda = 0.61806$ is not obvious, but however when $\lambda = 0.619$, the curve lifts off the x-axis. We are currently investigating this phenomenon.

References

- Box, G.M. and Jenkins, G.M. (1976). *Time Series Analysis: Forecasting and Control*. Holden-Day.
- Feller, W. (1971). *An Introduction to Probability Theory and Its Applications*. John Wiley & Sons, Inc.



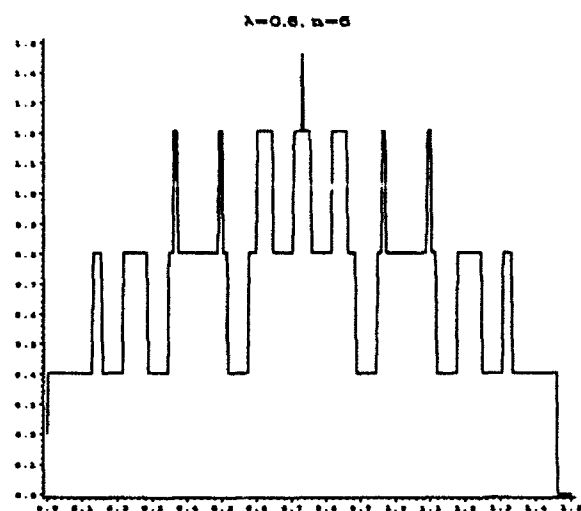


Figure 12
 $\lambda=0.6, n=10$

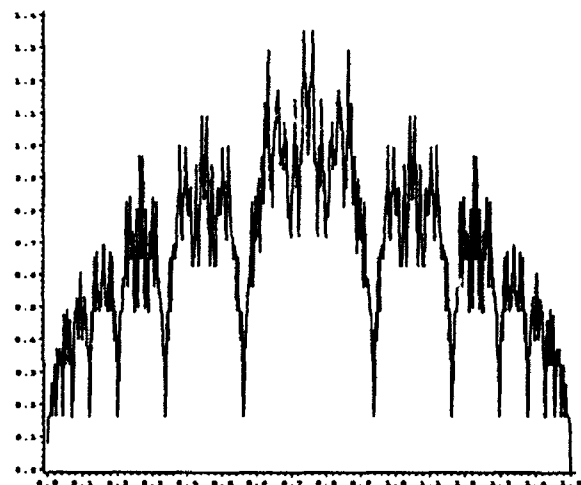


Figure 13
 $\lambda=0.617$

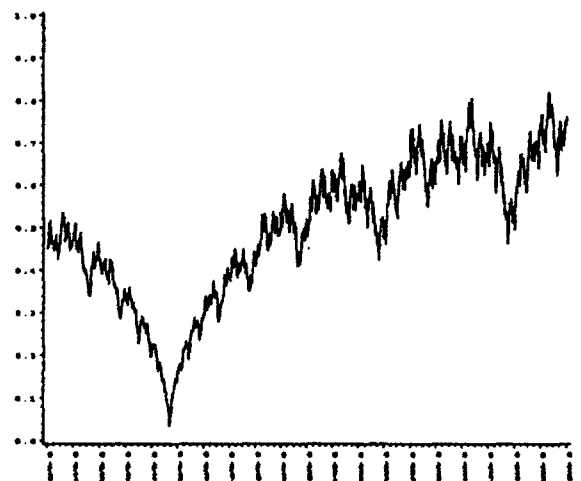


Figure 14

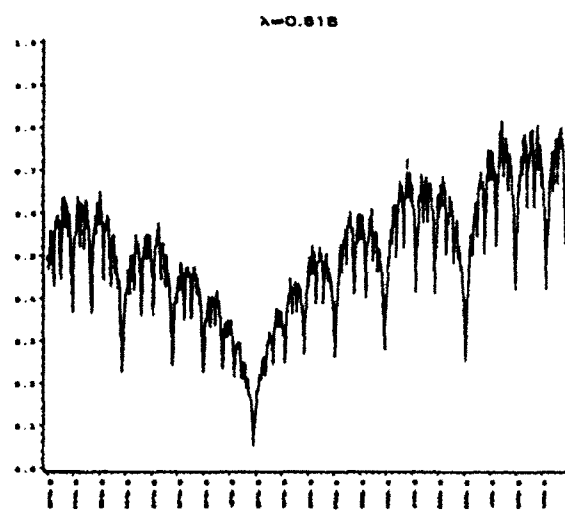


Figure 15
 $\lambda=0.61806$

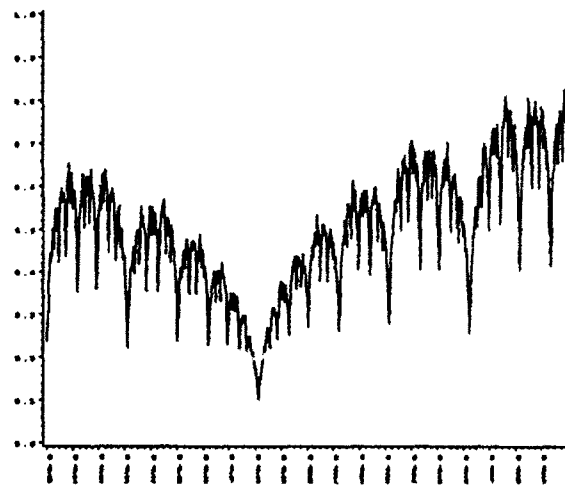


Figure 16
 $\lambda=0.619$

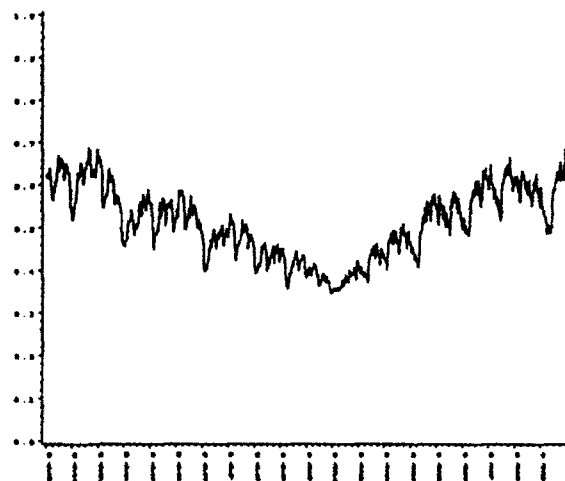


Figure 17

Visualizing the embedding of objects in Euclidean space

Michael Littman, Deborah F. Swayne, Nathaniel Dean, and Andreas Buja

Bellcore

Morristown, NJ 07962-1910

ABSTRACT

Matrices representing dissimilarities within a set of objects are familiar in mathematics, statistics and psychology. In this paper we describe XGvis, a software system which accepts diverse input data, such as graphs and multivariate data, develops a dissimilarity matrix from the data, and then iteratively and interactively embeds objects in a Euclidean space of arbitrary dimension. Using a technique called multidimensional scaling, objects are positioned so that their pairwise distances match the target dissimilarities as well as possible. Users can interact with XGobi, a software system for visualizing high-dimensional data, to browse the resulting embeddings. Mathematicians and statisticians have found XGvis to be useful for discovering and exploring structure.

XGvis runs under the X Window SystemTM.

1. Introduction and motivation

A dissimilarity matrix D is a matrix in which $D_{i,j}$ represents the degree of dissimilarity between each pair of objects i and j . The row objects and the column objects are usually the same and dissimilarity is assumed to be a symmetric relation. Matrix D is then a square symmetric matrix, $D_{i,j} = D_{j,i}$, with zeros on the diagonal and non-negative entries everywhere else.

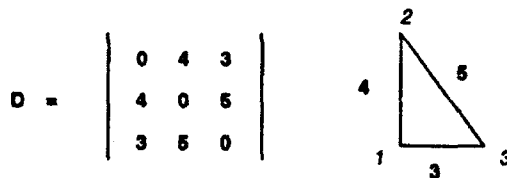


Figure 1: A dissimilarity matrix and its graphical interpretation.

The matrix in Figure 1 tells us how dissimilar 3 objects are from one another. Object 1, for instance, differs from object 2 by a score of 4. Construing similarity as

closeness and dissimilarity as distance gives D a natural graphical interpretation. The Euclidean distances among three objects arranged in the right triangle shown in Figure 1 precisely match the dissimilarity matrix.

In general, a geometric interpretation for a dissimilarity matrix D is created by representing each object as a point in some k dimensional space such that the distance between two objects i and j approximates $D_{i,j}$, well as possible.

The process of creating these graphical representations is known as multidimensional scaling (MDS) and has been studied for many years. Recently, interactive tools for visualizing these representations have become accessible. In this paper, we describe a system called XGvis which combines procedures for computing and displaying distance embeddings with a collection of functions for generating input dissimilarity matrices from several data sources.

How might such a system be useful? Dissimilarity information is very common in mathematics, statistics and psychology. Potential sources include shortest path distances among vertices in a graph, measured dissimilarity among test objects in a psychometric experiment, and distances among cases in multivariate data.

The use of XGvis for graph theory application is driven by another visualization tool developed at Bellcore. NETPAD (Dean et al., 1991) is an interactive program for creating graphs and graph layouts and analyzing these structures with a library of algorithms. NETPAD is capable of producing many interesting two-dimensional layouts, and when NETPAD and XGvis are used together, they can create and display layouts of graphs in three or more dimensions. To the degree that a higher dimensional layout better captures interesting aspects of a graph's structure, it can lead to special insights into properties of the graph. These insights can be helpful in a mathematician's search for conjectures and counter-examples.

Dissimilarity information can also serve as an intermediate step between a high dimensional collection of data and a lower dimensional view of it. By computing the distance between pairs of points in a ten dimensional space and fitting these distances with objects in a three dimensional space, one can often get a more intuitive

TM X Window System is a trademark of MIT.

feel for the original data. This process is called dimension reduction and is widely used in data analysis and visualization. XGvis is well suited for this task.

There are some cases when a dissimilarity matrix itself is the raw data to be viewed. It is a fairly common technique in experimental psychology to extract similarity judgments from human subjects about a collection of objects. These similarity judgments are easily transformed to dissimilarities and viewed using XGvis.

By molding diverse input data into the form of a dissimilarity matrix and creating graphical views of the dissimilarity information, XGvis has proven to be a powerful tool for people in many disciplines.

2. The Method

XGvis accepts various types of input data and develops a dissimilarity matrix from each. It then uses a form of multidimensional scaling to iteratively determine an embedding for the data.

2.1. Handling of different data types

XGvis handles three primary data types: graphs, multivariate data and dissimilarity judgments. A dissimilarity matrix is developed from each of these using either a default method or one chosen by the user.

A graph consists of a set of objects called vertices and edges where vertices can be represented as points and edges as lines connecting pairs of points. For example, vertices could represent cities and edges could connect pairs of cities that have a direct flight between them. Graphs are used to represent such diverse systems as telephone networks, molecules and patterns of bibliographic references.

XGvis accepts graphs in several forms. The simplest is as a list of edges. Edges in XGvis are undirected – that is, if i is connected to j then j is assumed to be connected to i . A user can also optionally indicate a set of initial locations for the vertices. If these are omitted, vertices are placed at random.

The default approach for creating a dissimilarity matrix from the graph is to compute, for all pairs of vertices in the graph, the number of edges on the shortest path between them. XGvis uses a dynamic programming approach for finding all the shortest paths quickly.

When multivariate data is passed to XGvis, a dissimilarity matrix is created by computing the Euclidean distance between each pair of cases in the raw data. The $D_{i,j}$ cell of the matrix is the distance between case i and case j . Non-Euclidean metrics are computed at the user's request.

XGvis can accept a dissimilarity matrix directly. These matrices are assumed to be symmetric. In the event that a non-symmetric (though square) matrix is passed in, XGvis' default action is to create a symmetric matrix from it by computing the elementwise mean between the matrix and its transpose. The resulting matrix is symmetric and can be used by multidimensional scaling for creating a graphical layout.

2.2. Multidimensional scaling

Multidimensional scaling has a long and interesting history. The defining goal of multidimensional scaling (MDS) is to take an $n \times n$ dissimilarity matrix D and produce vectors x_1, \dots, x_n such that the computed distance between every x_i and x_j approximates $D_{i,j}$. Up to 1964, people used classical or Torgerson-Young MDS (Torgerson, 1958) in which the dissimilarities are converted to an inner-product matrix and an eigenanalysis is performed.

In the early 60s, an alternate view of MDS began to emerge. The seminal contributor was Shepard (1966) who showed that to construct a faithful embedding is sufficient to know the ranks rather than the actual values of the dissimilarities. That is, if for every two pairs (a,b) and (c,d) of objects it is known whether a is closer to b than c is to d , then a faithful embedding can be constructed. Shepard gave evidence that this was possible and Kruskal (Kruskal, 1964a, 1964b) devised an optimization criterion which incorporates this idea. Kruskal defined a measure of fit called "stress" (S).

$$S(D, x, f) := \frac{\sum_{i,j} [f(D_{i,j}) - \|x_i - x_j\|]^2}{\sum_{i,j} \|x_i - x_j\|^2} \quad (2)$$

Given a current configuration of the objects x_1, \dots, x_n , the target dissimilarity matrix D and a monotonically nondecreasing function f , S is a measure of how poor the current embedding is. It is a normalized residual sum of squared differences between each $f(D_{i,j})$ and the distance from x_i to x_j . Kruskal provided a procedure for minimizing the stress by computing partial derivatives and applying the classical steepest descent method.

Kruskal's MDS based on this criterion with optimization of f over all monotonically nondecreasing functions is called "nonmetric MDS." It is called "nonmetric" since the actual values $D_{i,j}$ are thrown away by transforming them, while their ranks and only their ranks are retained because of monotonicity of the function.

The version of MDS used in XGvis is a metric relative of Kruskal's MDS. It optimizes x but leaves the function

f for the user to choose interactively from the set of powers of D .

Solving for the x_i 's in metric Kruskal MDS is a sticky non-linear optimization problem. It does not boil down to an eigendecomposition such as the one used in Torgerson-Young MDS. Metric Kruskal is a more powerful method than classical Torgerson-Young scaling.

Unlike performing an eigenvalue decomposition, there does not seem to be a closed form algorithm for finding a metric Kruskal MDS embedding. Instead we use an iterative algorithm for finding the x_i 's which minimize S .

The algorithm starts with some initial layout, which may be random, and minimizes S through a series of gradient descent moves on the point locations. With each step, the program moves every point a small amount in the direction which causes the stress to decrease maximally.

Aside from problems with local minima, easily avoided by using a large step size, the points settle to a reasonable layout quickly.

Visualizing an iterative algorithm offers two main benefits. First, it is possible to observe changes in the optimization behavior that might result from changes in the parameters of the stress function. For instance, one could experiment with an additional penalty term for constructing constrained layouts. Second, a user can find a good layout visually without having to construct a stopping criterion such as "run for x steps" or "run until the error drops below ϵ ." An iterative method allows a user to implicitly use the criterion, "run until it looks right."

We have found this particularly important for setting the step size parameter. Larger steps help MDS make fast progress early and skip over local minima. Later in the run, a smaller step size is needed to let MDS settle on a solution. Terms like *larger*, *later* and *smaller* are very much problem dependent but are easily determined (at least in 2 and 3 dimensional embeddings) by watching the points move.

Interactive MDS optimization was implemented by one of us on a workstation in 1982, but it was never documented other than in a film (Buja, 1982).

3. Control and Visualization

Embeddings in XGvis proceed as follows: a user begins an XGvis session by invoking the program with some initial data. Using default routines, XGvis creates a dissimilarity matrix from the data. The user is then presented with a graphical interface for controlling the progress of MDS and manipulating the view of the cur-

rent embedding.

Typically, the user will set some MDS parameters then "run" MDS. The points are visible in a display window while MDS moves them. When the user is satisfied with the layout, MDS can be stopped and the resulting embedding examined with an integrated version of the XGobi visualization program (Swayne et al., 1991c).

All interactions take place in two windows on workstation or personal computer screen. The XG control panel window is used to start, stop and reset MDS as well as to control the MDS-related parameters. The XGobi display window is used to watch the progress of the optimization and to interact in various ways with the view of the data or figure.

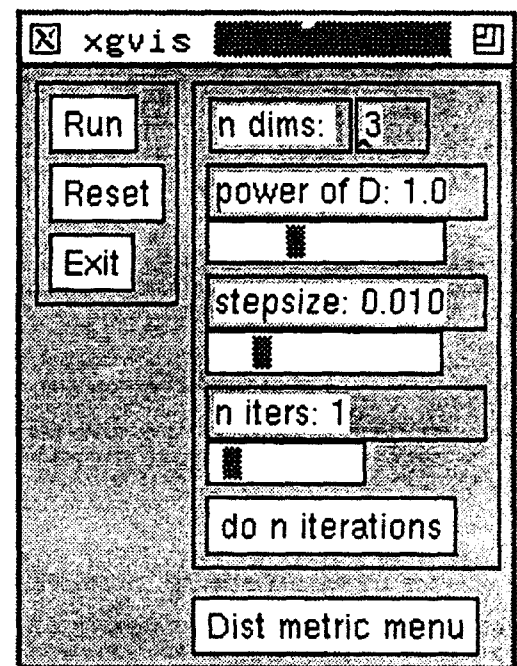
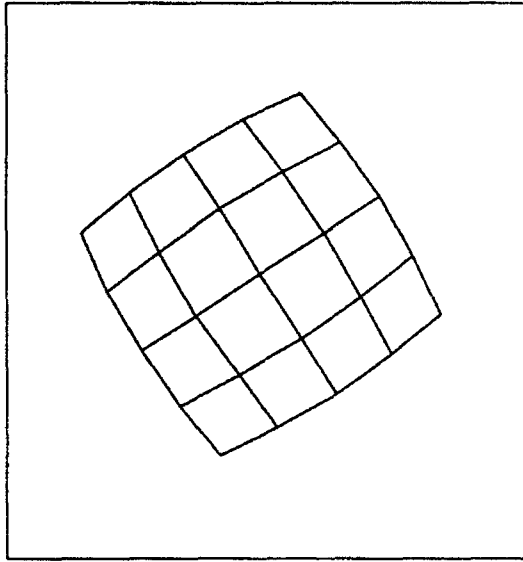
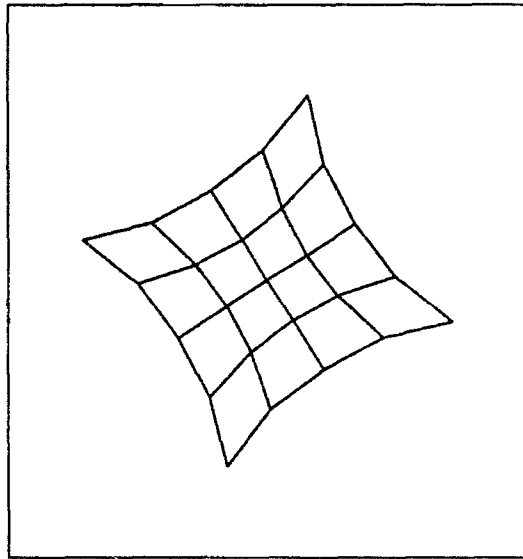


Figure 2: XGvis Control Panel

3.1. Control panel

The XGvis control panel is shown in Figure 2. The leftmost subwindow contains a *Run* button that allows the user to start and stop the optimization and a *Reset* button that returns the data or figure to its original configuration. The labels, buttons and scrollbars in the rightmost windows allow a user to adjust various optimization and scaling parameters. These parameters can be set before MDS begins or while it is running; they are briefly described here.

Figure 3: Grid, power of $D = 0.5$ Figure 4: Grid, power of $D = 2.0$

Dimension: By typing an integer into the text window at the right of the label $n\ dims$, the user specifies the dimension of the Euclidean space in which to embed the point cloud or figure. Reasonable choices are 2 (XGvis embeds the points or objects in the plane), 3 (XGvis constructs a three dimensional layout), or up to 5 or 6, in which case special techniques and imagination are

required to interpret the resulting structures. The interface actually permits embeddings in up to twenty dimensions but we have not found use for nearly that many. Occasionally one dimensional embeddings (all points on a line – unidimensional scaling) are interesting and these are also supported.

Transformation: The user adjusts the scrollbar beneath the label *power of D* to select the power to which each element of D should be raised. In this way, XGvis allows some variation in the function f shown in Equation 2.1. We restrict $f(D_{i,j})$ to $D_{i,j}^p$, where p is the value read from the scrollbar. Typically, a value of $p = 1.0$ is the most meaningful: this tells XGvis to fit the dissimilarities in D directly.

Striking and useful effects can be obtained by changing p from its default value. Small values of p tend to minimize differences in the dissimilarity matrix, often resulting in a “rounder” or more compact layout. A 25 node graph connected as a 5×5 grid appears perfectly square when displayed with $p = 1.0$, but note the rounded appearance of the 5×5 grid as it is shown in Figure 3: this embedding was produced using $p = 0.5$. In the limit $p = 0.0$ and MDS tries to make all pairwise distances 1 from each other. See Buja et al. (1991) for a mathematical treatment of this “null” case.

Larger values of p (with 3.0 being the highest setting on the scrollbar) have an opposite effect. Differences in dissimilarity are accentuated resulting in more “spiky” pictures. Figure 4 demonstrates the effect of setting $p = 2.0$ for the 5×5 grid graph.

Step size: The MDS update rule involves moving each object in the direction of the gradient of Equation 2.1. The *stepsize* scrollbar controls the size of the step in the direction of the gradient. Larger values cause the embedding to proceed rapidly, often skipping over dangerous local minima. However, they run the risk of causing the embedding to “thrash,” that is, jump out of control from one configuration to another. Smaller values are useful when a figure has begun to settle.

Iteration control: With $n\ iters$ set to 1, XGvis will update the graphics window after every MDS step. For more complex embeddings, it is sometimes more efficient to take several MDS steps before redrawing. Larger values of $n\ iters$ accomplish this. The *do n iterations* button allows a user to perform a fixed number of MDS steps.

Distance metric: The user can bring up a menu of routines for recomputing the target dissimilarity matrix. The metrics implemented include shortest path distances for graphs and Euclidean and Manhattan distances for multivariate data. This function is primarily useful for experimenting with different types of dimension reduction methods or in cases where there is ambiguity as

what the default dissimilarity matrix should be.

3.2. Visualization with XGobi

When the optimization is running, the plot displayed in the XGobi window is redrawn with each iteration so that the motion of the points can be observed. If the data under study is a graph, setting the figure in motion while the optimization proceeds is particularly helpful. Using XGobi's *Rotate* or *Tour* mode, a viewer sees a smooth sequence of two-dimensional projections of the changing figure. Viewers describe this as watching the figure "unfold" or "puff out."

Once the user decides that the figure or point cloud has become stable, many of XGobi's other interactive functions can be used to advantage. Selecting the *Scale* mode allows the figure to be stretched or reshaped on the screen using mouse motions: this is especially useful if the viewer wants to zoom in on a subsection of a complex figure. In *Brush* mode, a user can interactively change the color of selected lines or points or change the shape and size of the plotting character used for selected points. Using *Identify*, a user moves the cursor near a point of interest, and its label appears. The label can be any user-supplied text string; by default, it is simply an identification number, 1 to n (the number of data objects). Point brushing and identification are linked: that is, if a user is working on the same data in another XGobi window (which could be part of another XGvis session or an independent instance of XGobi), then a point that is brushed or identified in one window is simultaneously affected in the second.

Some useful ways of selecting subsets by brushing and rerunning MDS on the brushed subsets were provided in an earlier implementation (Buja, 1982). These have not been included in the current XGvis system.

XGvis calls XGobi as a function (Swayne et al., 1991). In this use of XGobi, its data structures are defined by XGvis, the parent program, and yet all the interactive methods of XGobi are available for use on that data.

Figures 3 - 12 in this paper were produced using an XGobi facility which enables the user to create a PostscriptTM representation of the contents of the plotting window.

4. External Interfaces: NETPAD

Command line arguments allow a user to call XGvis in several ways: with a dissimilarity matrix, point locations, a graph specified as edges and point locations or a graph specified in NETPAD format.

NETPAD is an interactive computer program for creating and analyzing graphs and graph algorithms. It contains a graphical user interface and an expanded toolkit of graph algorithms. It can be used like an electronic pencil and notepad to create, modify, save, call and delete graphs and their associated attributes or attribute values. NETPAD has a library of programs for manipulating and analyzing graphs. These programs can be predefined functions which are part of NETPAD or user-defined programs such as XGvis which are interfaced to NETPAD.

All of the graph layouts displayed in NETPAD windows are two dimensional, and it is this limitation that XGvis enables it to surmount. Users can create and manipulate graphs using NETPAD and smoothly pass them to XGvis to be embedded in a higher dimensional space and then viewed using interactive motion graphics.

5. Examples

In this section, we provide two in-depth examples of applications of the XGvis system. They were chosen to demonstrate the range of possible uses for the system from abstract mathematics to data analysis. The first involves using XGvis to construct and view a three dimensional layout of vertices for a well-known graph. In the second example, we use XGvis to find structure in empirical confusion data.

5.1. Adjacent Transposition Graph

Creating views of graph structures can be helpful to discrete mathematicians who are generating conjectures about attributes of a mathematical system. Graphs are useful in statistics for representing relationships between data measurements (Thompson, 1991). A good picture of a graph can make important trends in the data more apparent.

Creating higher dimensional views of graphs was one of the primary motivations for embarking on the XG project. We identified numerous methods for moving from a set of edges and vertices to a set of spatial locations for those vertices. We implemented and studied several of these including spring embeddings, isometric decomposition (Winkler, 1987), and a higher dimensional generalization of the Tutte embedding (Tutte, 1963).

Most of our efforts were focused on the multidimensional scaling algorithm described in this paper. Our embedding strategy was to make an analogy between shortest path distance in the graph and Euclidean distance in the resulting embedding. Starting with a graph represented as a list of vertices and edges, we compu

TMPostscript is a trademark of Adobe Systems, Inc.

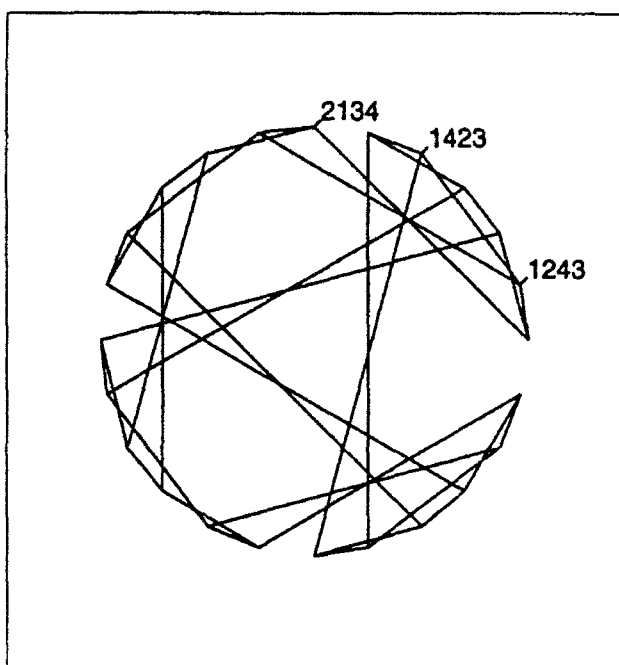


Figure 5: Adjacent transposition graph, first position

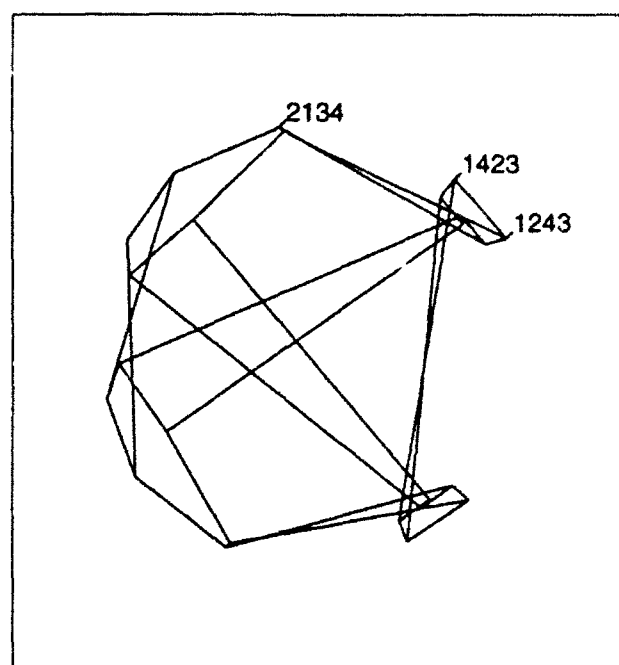


Figure 6: Adjacent transposition graph, unfolding

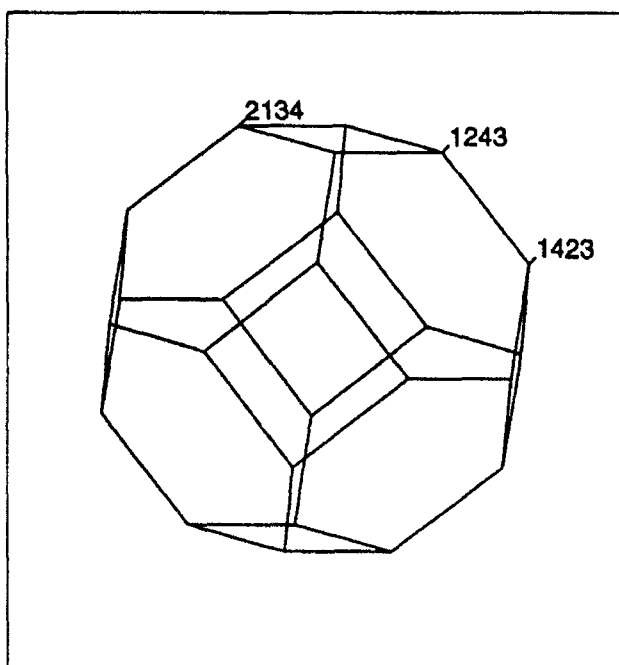


Figure 7: Adjacent transposition graph, final position

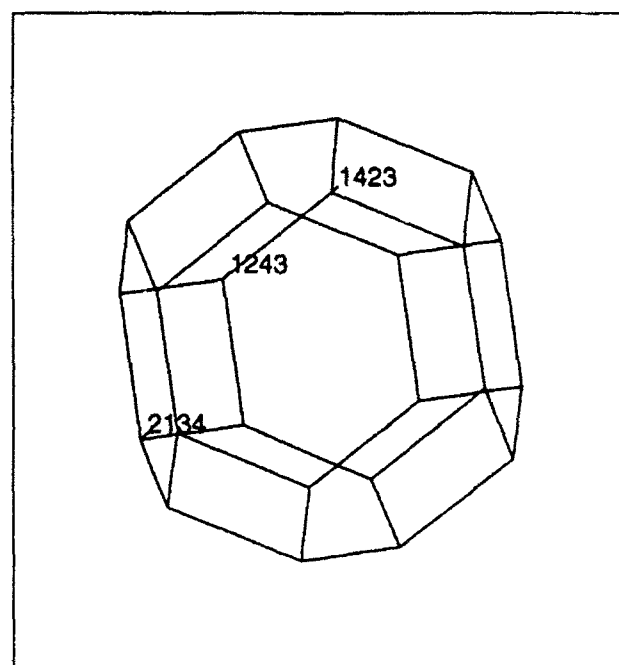


Figure 8: Adjacent transposition graph, rotated view

using dynamic programming, a matrix D in which D_{ij} is the number of edges along the shortest path between vertex i and vertex j in the original graph.

We start with some initial assignment of vertices in the graph to positions in k dimensional Euclidean space; this initial mapping could simply be random. Then we use iterative multidimensional scaling to move the vertices to positions that better reflect the shortest path distances in D . Larger k 's make it possible for MDS to fit the data better while smaller k 's give humans a better chance of interpreting the resulting layout. The most instructive pictures involve trading off these forces and choosing the smallest k that appears to capture the significant structure in D . Often this involves a bit of trial and error.

Let us consider a graph, shown in Figure 5, known as the $n = 4$ adjacent transposition graph. This graph has special meaning to discrete mathematicians. It was passed on to us by Winkler, who was using it to study extensions of partial orderings. Thompson (1991) has also used this graph to depict relationships between surveyed preference judgments. For exploratory and explanatory purposes, it is useful to have a drawing that shows nodes as close together if and only if they are close together in the actual graph.

The graph is generated as follows. We start with all permutations of the sequence 1,2,3,4. There are $4!$ or 24 such sequences and we make each a vertex in the graph. We connect two vertices by an edge if one permutation can be turned into the other by simply transposing two adjacent elements. Figure 5 shows an initial layout for this graph; three of its nodes are labeled with their corresponding permutations. Observe that the node corresponding to 1423 is connected to 1243 because one can be turned into the other by swapping the middle two digits. However, 2134 is not connected to 1423 because there is no way to move from one to the other with a single adjacent flip. How many flips would it take? What we'd like is a representation of the graph that makes that easier to see.

Figures 6, 7 and 8 show XGvis finding the optimal three dimensional Euclidean layout for this graph, with the power of D set to its default value of 1.0. Figure 6 shows an intermediate stage in the "unfolding" of this graph, and Figures 7 and 8 show two views of the final position determined by the MDS algorithm.

A three dimensional embedding clarifies the structure a great deal. The resulting shape is equivalent to a truncated octahedron. One striking feature of the graph is its collection of six cycles and four cycles – hexagons and squares. Each six cycle is the collection of all permutations in which one end is held fixed and all permutations

of the remaining three digits enumerated. Each square consists of the permutations which have the same symbols in the first two positions and the last two positions.

What was invisible is now clear. We can see in Figure 7 and 8 that 2134 and 1423 are just 3 flips apart and that there are two paths of that length. Interestingly the sequence 1243 lies on both of them. The pair of nodes corresponding to 2134 and 1243 are both on single square face. This face consists of all sequences which start with 1 and 2 and end with 3 and 4. The pair 1243 and 1423 are together on 2 hexagonal face one represents all sequences ending with 3 and the other all those starting with 1.

We have used XGvis to examine other graphs from this family including $n = 3$ ($3! = 6$ nodes connected as a hexagon) and $n = 5$ ($5! = 120$ nodes which require 4 spatial dimensions to draw sensibly and is therefore difficult to visualize).

5.2. Morse code confusion data

In the previous example, the dissimilarity matrix that XGvis derived was based on the structure of a graph. Now we experiment with fitting an explicit dissimilarity matrix.

In Figure 9, each point represents a Morse code sequence – one for each of the symbols a-z and 0-9. A matrix of *confusion rates* for these symbols was determined in a psychological experiment by Rothkopf (see Gnanadesikan, 1977, pg. 44ff for an accessible account). To prepare the data for XGvis, we needed to convert the confusion rates – the percentage of times one symbol was mistaken for another – to dissimilarities. This was accomplished by subtracting each of the confusion rate from 1. The resulting matrix is recognized by XGvis as asymmetric and is made symmetric by the transformation described in Section 2.1. It has been discovered empirically that a good layout is found for this data in two dimensions. This is especially true if the dissimilarity matrix is transformed by cubing each cell; i.e., let $f(D_{ij}) = D_{ij}^3$.

Starting from a random layout, as shown in Figure 9, the points gradually migrate to positions that better approximate the target dissimilarity matrix. With the user watching in anticipation, they soon sort themselves out. The final layout is shown in Figure 10. It has been augmented by hand with some information useful for interpreting the result.

Figure 11 contains an XGobi plot of the percentage of dots used to represent each symbol versus its length in characters. (Recall that symbols are represented in Morse code by a sequence of one to five dots and/or dashes.) The plot in Figure 11 has been brushed along

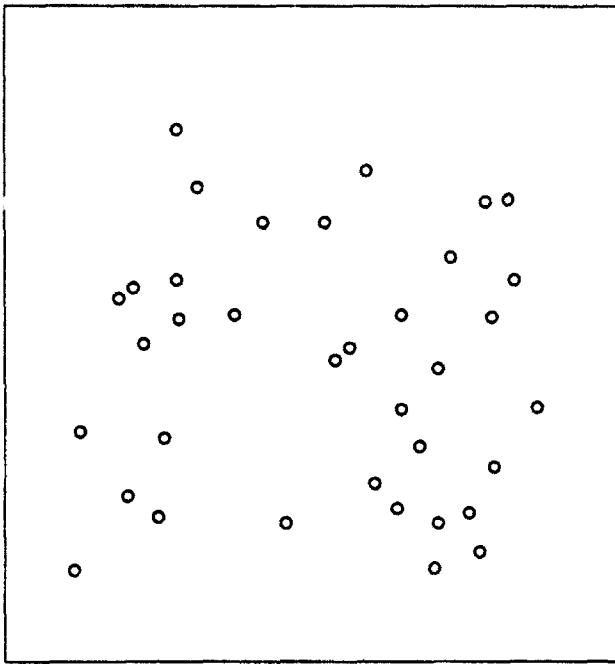


Figure 9: Morse code data, initial positions

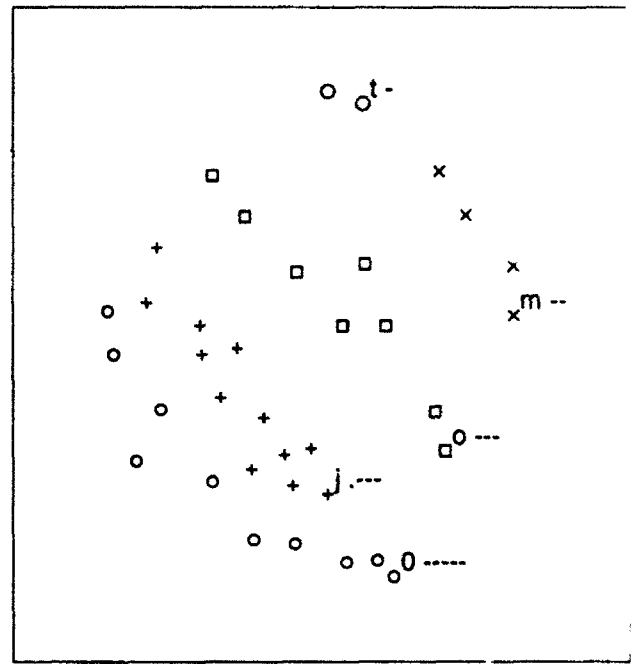


Figure 10: Morse code data, final positions; here the plot is brushed by the length of the Morse code representation of the symbol.

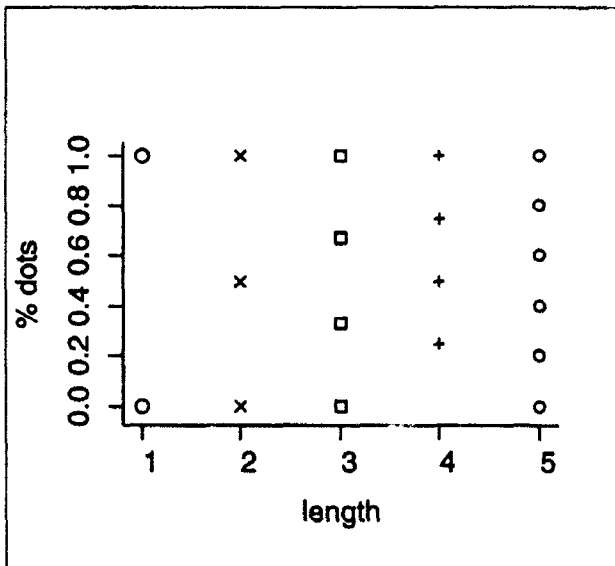


Figure 11: Morse code data, XGobi plotting window

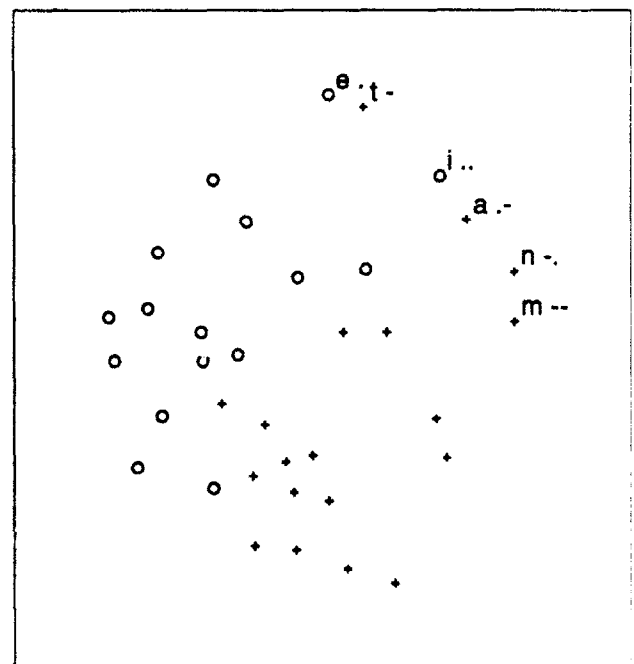


Figure 12: Morse code data, final positions; here the plot is brushed by the percentage of dots in the Morse code representation of the symbol.

the horizontal axis so that symbols with the same length representation in Morse code are drawn with the same glyph; for example, symbols represented by two characters (specifically i, a, n and m) are plotted using an "x".

The XGobi plot of % dots versus length in Figure 11 is linked to the XGvis plot in Figure 10, so the points in Figure 10 are similarly brushed. One symbol of each length has been labeled: the letters t, m, o, j and the number 0. Note that in the MDS representation of the confusion data, points form groups based on the length of the corresponding Morse code sequence.

Within each clump, the symbols have arranged themselves so that Morse code symbols with a high fraction of dots are above and to the left while those with relatively more dashes are below and to the right. This is shown in Figure 11, which was created by once more brushing the XGobi plot, this time along the vertical axis. The one-character and two-character symbols are all labeled: e (one dot and thus 100% dots) and t (one dash, 0% dots) are now brushed differently; i (two dots, 100% dots) is brushed differently than a and n (dot-dash and dash-dot, 50% dots) and m (two dashes, 0% dots).

Information such as this can be of particular use to code designers. By understanding the patterns of confusion, it might be possible to generate a new code that is less prone to misinterpretation.

6. Conclusion and Extensions

XGvis provides, in a single program, modules for creating, embedding, and displaying dissimilarity data. It is remarkably general in scope, providing useful functionality to mathematicians, statisticians and psychologists. The integration of computation and display make using the system natural; useful work can be accomplished by someone who has no expert knowledge of multidimensional scaling.

We have planned to extend the XGvis system with several other embedding algorithms. Nonmetric MDS (described in Section 2.2) has been used successfully for embedding graphs (Fairchild et al., 1988). Extensions are possible for drawing directed graphs – for instance, constraining edges to point down and to the right if possible. It is also possible to add additional terms to Equation 2.1 to encourage MDS to lay out the objects within a certain rough outline. This can be important for graphical layout applications.

The XGvis system described in this paper is not publicly available at the time of this writing. However, the XGobi and NETPAD systems on which it is based are available. Contact the authors for de-

tails (dfs@bellcore.com for XGobi; nate@bellcore.com for NETPAD).

7. Acknowledgments

We'd like to thank Paul Tukey for taking the trouble type in the Morse code confusion data.

References

- Borg, I. and Lingoes, J. (1987). *Multidimensional similarity structure analysis*. Springer-Verlag.
- Buja, A. (1982). MDS in an interactive environment. Half hour film, ORION project, Stanford Linear Accelerator Center.
- Buja, A., Logan, B. F., Reeds, J. R., and Shepp, L. (1991). Inequalities and positive-definite functions arising from a problem in multidimensional scaling. Bellcore and AT&T Bell Laboratories Technical Memoranda.
- Dean, N., Monma, C. L., and Mevenkamp, M. (1991). NETPAD User's Guide. Bellcore Technical Memorandum.
- Fairchild, K. M., Portrock, S. E., and Furnas, G. V. (1988). SEMNET: Three-Dimensional Graph Representations of Large Knowledge Bases. In J. Guindon, R., editor, *Cognitive Science and Its Applications for Human Computer Interaction*, pages 201-233. Lawrence Erlbaum, Hillsdale, New Jersey.
- Gnanadesikan, R. (1977). *Methods for Statistical Data Analysis of Multivariate Observations*. John Wiley and Sons, New York.
- Kruskal, J. B. (1964a). Multidimensional scaling by optimizing goodness of fit to a nonmetric hypothesis. *Psychometrika*, 29:1-27.
- Kruskal, J. B. (1964b). Nonmetric multidimensional scaling: a numerical method. *Psychometrika*, 29:115-129.
- Kruskal, J. B. and Wish, M. (1978). *Multidimensional Scaling*. Sage Publications, Beverly Hills.
- Shepard, R. N. (1962). The analysis of proximities: multidimensional scaling with an unknown distance function, I and II. *Psychometrika*, 27:125-139 and 219-246.

- Swayne, D. F., Buja, A., and Hubbell, N. (1991a). XGobi Meets S: Integrating Software for Data Analysis. In *Proceedings of the 23rd Symposium on the Interface*. Interface Foundation of North America, Inc.
- Swayne, D. F., Cook, D., and Buja, A. (1991b). User's Manual for XGobi, a Dynamic Graphics Program for Data Analysis Implemented in the X Window System (Version 2). Bellcore Technical Memorandum.
- Swayne, D. F., Cook, D., and Buja, A. (1991c). XGobi: Interactive Dynamic Graphics in the X Window System with a Link to S. In *1991 Proceedings of the Section on Statistical Graphics*. American Statistical Association.
- Thompson, G. L. (1991). Exploratory Graphical Techniques for Ranked Data. In *Proceedings of the 23rd Symposium on the Interface*, Fairfax Station, VA. Interface Foundation of North America, Inc.
- Thompson, G. L. (1993). Generalized Permutation Polytopes and Exploratory Graphical Methods for Ranked Data. *Annals of Statistics*. To appear.
- Torgerson, W. S. (1958). *Theory and Methods of Scaling*. John Wiley and Sons.
- Tutte, W. T. (1963). How to draw a graph. *Proceedings of the London Mathematical Society*, 13:743-767.
- Winkler, P. (1987). The metric structure of graphs: theory and applications. In Whitehead, C., editor, *Surveys in Combinatorics*, pages 197-221. Cambridge University Press.

Statistical Visualization

John Sall, Sr. Vice President, SAS Institute

Every statistical fit is the result of a balancing and optimization process. That is one of the keys to bringing a visual understanding to statistical methods. The balancing and optimization help one understand the picture by translating visual displacements and distances into conceptual forces and energies. The hypothesis leverage plot, the comparison circles plot, and the contingency mosaic plot are good examples of graphs that are consistent with mechanical conceptual models.

This paper condenses material from Sall (1989, 1990, 1991a, 1991b, 1991c and 1992). Some of the ideas are also covered by Farebrother (1987).

Visualization with Inner Vision

Visualization is not just what you see, but what you envision as the conceptual mechanism to recognise and understand what you see.

Imagine that there is a little man in your head that watches television, i.e. he watches the images from your retina and tries to make sense of them. Most of the time it is pretty dull material. His job is to try to match what he sees with what he remembers in his cognition memory, and find important features.

In discussing visualization, one of the most important neglected topics is how to consider and train this inner vision. Much has been written about how to make statistical graphs neat and informative, but little about how to approach our inner vision. Cognitive scientists like Don Norman do think about such things. They worry about why most people don't learn how to program their VCR's. They help explain why people find the Macintosh so much easier to use than other computers. The principal lesson of their work is that the key to good user interface is to follow a familiar *conceptual model*. The conceptual model is the metaphor that our inner vision uses to make sense of what we see. Without a conceptual model, graphs are just a collection of pixels, and statistics is just a bunch of mathematics.

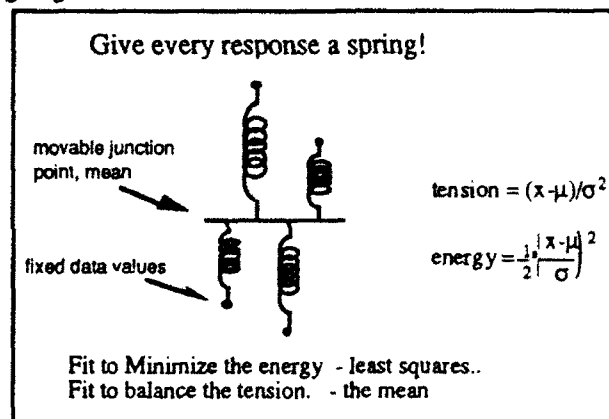
Force and Energy

With a conceptual model, you can visualize qualities that are not directly seen. I think that the most useful qualities are *force* and *energy*. So in educating our inner vision, we should seek mechanical models that translate statistical concepts into forces and energies.

Springs

Let's take the statistics of normal-theory least-squares fit and find the appropriate mechanism. We need a mechanism that costs energy as the square of the distance. With this we can equate least squares with minimum energy. We need a mechanism that exerts resistance with force linear to displacement. We need a mechanism that can equate the linear algebra principals of orthogonality to the mechanical notion of balance.

Let's take the simplest case, fitting a mean to a batch of values. Get a spring for each point, connecting one end to the point, and the other end to a movable junction that is going to be used to estimate and test means.



Estimation-least squares

The best estimate of the mean is the junction point of the springs where least energy is stored.

Estimation-orthogonality

The best estimate of the mean is the junction point where the forces pulling it from above and below balance.

Testing Hypotheses

To test the hypothesis that the mean is some value, move the junction point to that value, and record how much additional energy it took to do that.

Sample Size

Add more points with more springs and the mean estimate is held more securely. It is easier to test hypotheses.

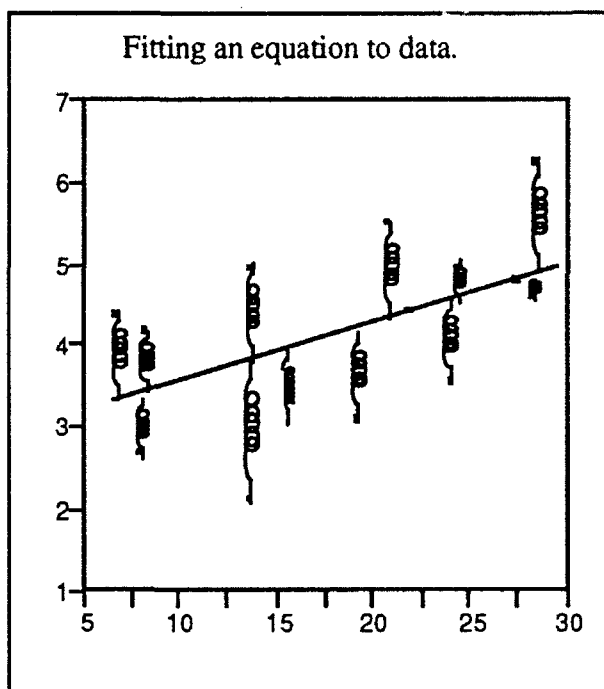
Error Variance

If you use springs with a higher spring constant $1/\sigma^2$ (smaller variance), the mean is held more securely. It is easier to test hypotheses.

Simple Regression

If you are fitting a line to a scatter of points, the same idea works. The line is balanced both up-and-down and

rotationally to estimate the intercept and slope simultaneously. You can test that the slope is zero by moving the line to a horizontal position at the mean, and recording the energy needed to do that.



Robust estimation

Since springs exert a huge force when a point gets far away, it might be preferred that a different mechanism be used. For example, you could envision a spring with a gradual release mechanism if it is stretched past a limit. Or you could use strings and weights instead (L1 estimates) with strings attached to the line, and led through holes at the points, with weights pulling on the string. (Farebrother, 1987)

Correlation

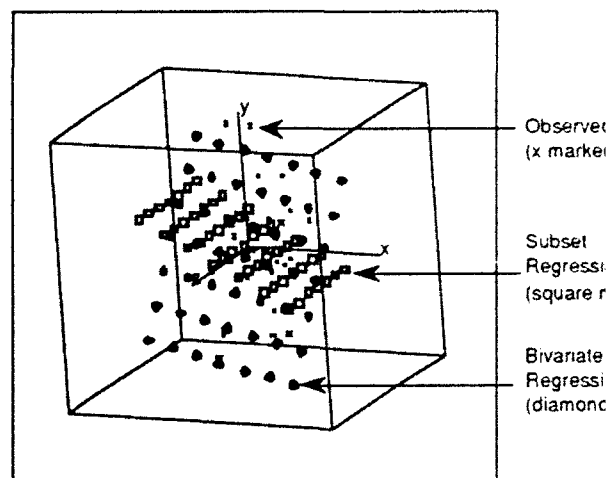
If the springs were not constrained to go vertical, then the balance line of minimum energy would be the first principal component. If the scale were made so that the balanced horizontal and vertical lines used the same energy, the this would be with respect to the correlation matrix. One minus the square root of the percent energy saved by the free line would be the correlation.

Multiple regression

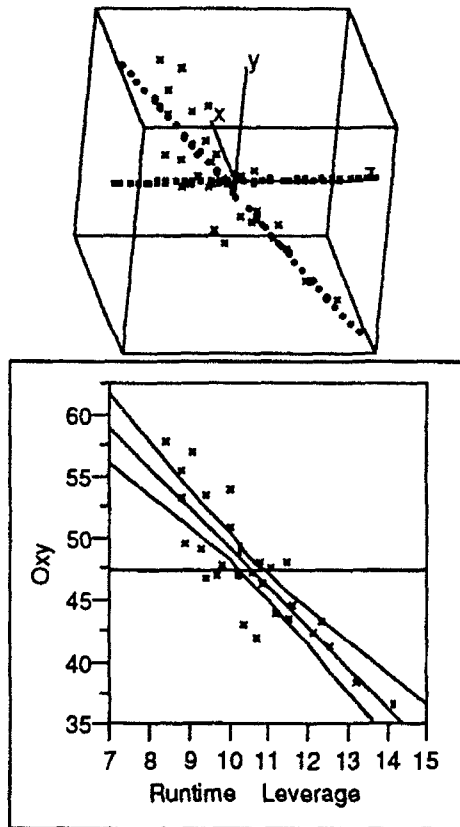
The same idea works in more dimensions for more regressors. For two regressors, you are fitting a plane to a bunch of 3D points with vertical springs.

Multiple Regression Tests

Tests in multiple regression are performed, as before, by comparing the total energy stored in the springs as you constrain a fit to the hypothesized parameter values while letting the remaining parameters free to seek a balance. In the following 3D graph, there are two fitting grids shown. One for the whole 2-regressor fit. The other results when one regressor (runtime) has been constrained to have parameter zero. Now imagine how much more energy needs to be stored in the springs of the constrained fit than the whole regression.

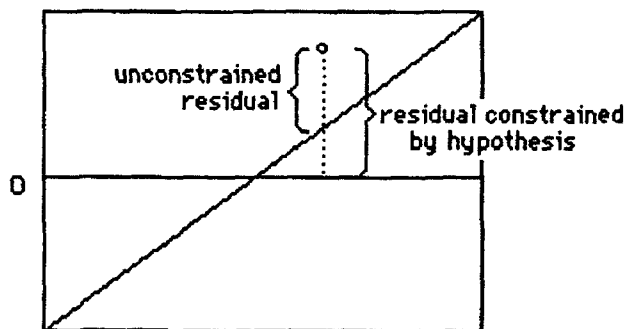


To get a nice perspective of the hypothesis test, spin the 3D plot so that you can see both fitting planes edge-on, and roll this so that the constrained plane is horizontal, shown in the first plot below. This is a very useful perspective, because you are reducing to 2-dimensions the problem of seeing a test in three dimensions. The residuals are the distance from each point to the plane. The test is just the difference between one sum of squared distances and the other. You can judge the distances even though the distances are somewhat tilted from the vertical representing the response. If you correct for the tilt, you get the perfect perspective of a test in a multiple regression, as shown in the second plot. Note that the points are nearly in the same position in both plots. The second plot is called an "hypothesis leverage plot".



Hypothesis Leverage Plot

It turns out that the idea of showing a multiple regression test can be generalized to any hypothesis test. You just apply the constraint representing the test and obtain the residuals from both the constrained and unconstrained fit. Then you plot the constrained-fit residual as the vertical position, and the difference as the horizontal position. You end up with a plot that can be interpreted just as it is with a simple regression.



With the dimensionality reduced to show the essential details of the test, you can now visualize what is going on. Now you can see point-by-point the composition of the test statistic. Now you can see the test as the spring energy metaphor in a reasonable number of dimensions. The test is how much energy it costs in the springs to

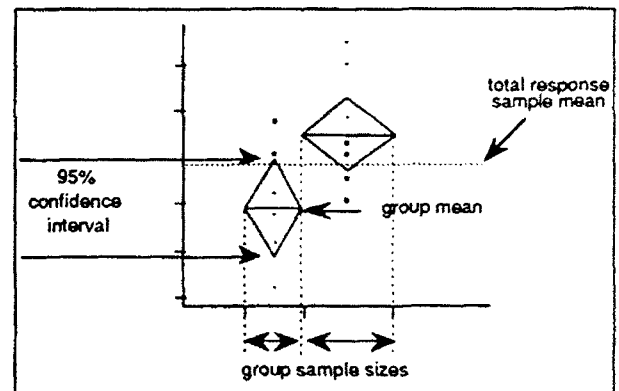
push the line of fit down to the horizontal line. Now you can appreciate the concept *leverage* as how far out a point is horizontally from the middle so that its springs force can have a leveraged effect on the test statistic. Changing the slope to the horizontal line will use up much more energy on the points that are farther out horizontally.

So we have the general leverage plot (Sall, 1990), a hypothesis-eyed view of the data. It is a generalization of the *partial-regression leverage plot* of Belsley, Kuh and Welsch (1980) or the *added variable plot* of Cook and Weisberg (1982).

One-way Anova, k Means

What conceptual mechanism is useful for comparing means in a one-way classification? What plot can be used to show if there are significant differences in the means?

The springs can be used again for the conceptual model. For each group in the classification, a vertical scatter of points can be shown, with imaginary springs suspending a different mean for each group. To test that the means are all equal, we constrain them to be the grand mean and measure the energy we needed to add to the springs.



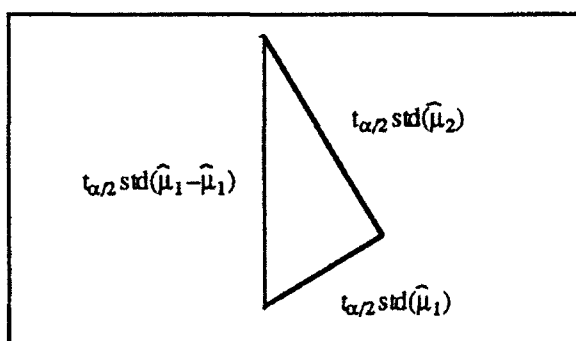
But how can we visualize comparing each mean with each other mean in one plot? That is hard. We start with a student's *t* test, which can be written as a formula for the distance such that if the means are farther apart than this least significant distance, they are significantly different.

$$LSD = t_{\alpha/2} \text{std}(\hat{\mu}_1 - \hat{\mu}_2)$$

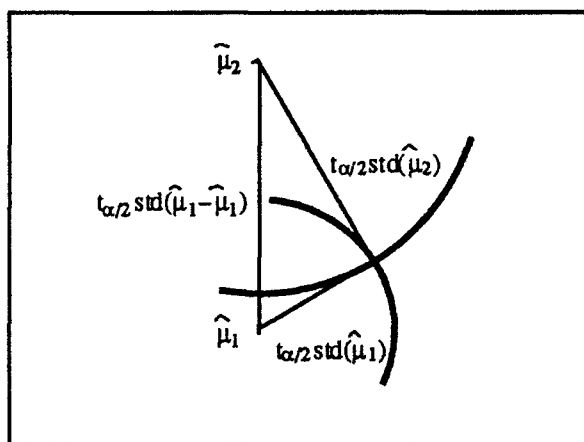
$$[\text{std}(\hat{\mu}_1 - \hat{\mu}_2)]^2 = [\text{std}(\hat{\mu}_1)]^2 + [\text{std}(\hat{\mu}_2)]^2$$

$$LSD^2 = [t_{\alpha/2} \text{std}(\hat{\mu}_1 - \hat{\mu}_2)]^2 = [t_{\alpha/2} \text{std}(\hat{\mu}_1)]^2 + [t_{\alpha/2} \text{std}(\hat{\mu}_2)]^2$$

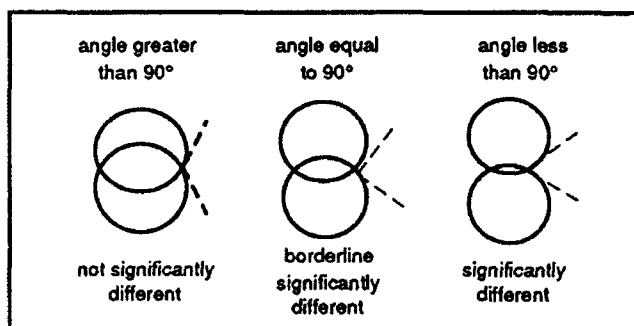
Now we look for something that we can use in the geometry of a plot. Notice that the LSD relationship is a square equal to a sum of two squares, which can be mapped into a Pythagorean triangle of standard errors of the means and their difference.



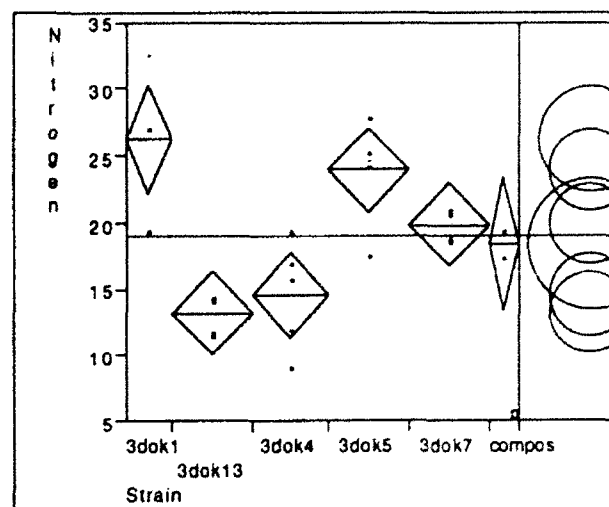
The hypotenuse gives us the LSD as measuring stick. Now suppose that two means are exactly different by the LSD and draw circles around each mean with radius equal to the 95% confidence interval for each mean. Geometrically, the circles must intersect at right angles.



Since the circles for two means that are different by the LSD intersect at right angles, we can make the rules for what happens if the means are separated by more or less than the LSD.



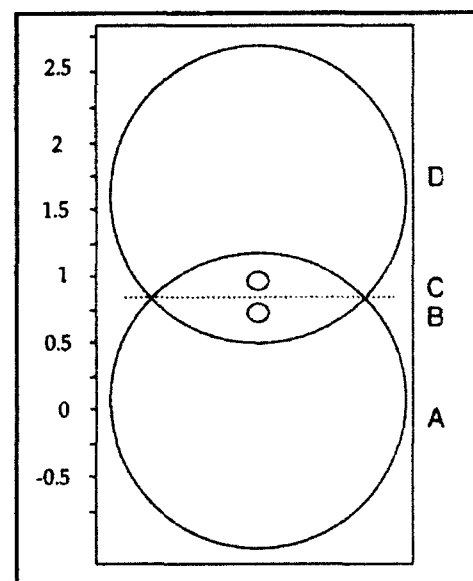
Applied to a set of data in a one-way layout, the circles are drawn in a vertical line, with centers at the means, and the radius taken from the 95% confidence interval. If circles do not intersect, the corresponding means are significantly different. If they nest, then they are not significantly different. If they intersect, then you look at the angle of intersection to determine if the difference is significant.



Smaller circles are "better" so that differences can be detected. The more springs (more observations), the more tightly bound the mean is.

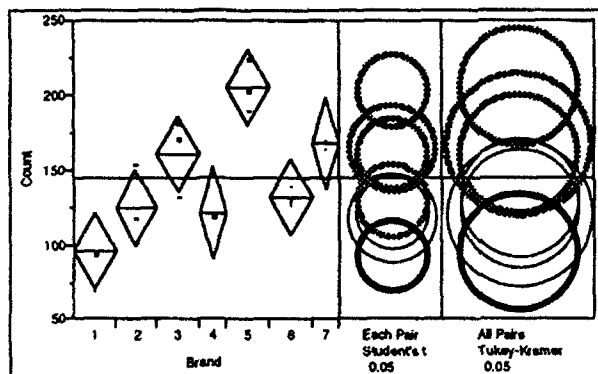
Sample Size: "How many observations do I need to drive the radius of the circles down in order that circles a given difference apart intersect at less than 90 degrees to become significantly different?"

Design: "The way to get the circles significantly separated with the minimum number of observations is allocate an equal number of observations to each of the two experimental conditions."



Strange, but true--two widely separated means from sma samples A and D are not significantly different. Two narrowly different means from large samples B and C are significantly different.

Comparison circles can be extended to cover multiple comparisons, such as the Tukey-Kramer HSD. The arithmetic is the same except for the HSD quantile instead of the student's t .



Categorical Response

Define *unexpectedness* to be the negative log of the probability attributed to the event. Expected unexpectedness is entropy. In statistical fits, the goal is to choose the fit to minimize the unexpectedness:

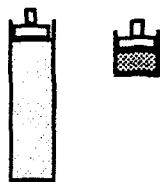
$$H = \sum_{i=1}^n -\log p_{Y_i}$$

where Y_i is the response that occurred.

For continuous normal responses, the log of the density takes the form of a sum of squared residuals, and the conceptual model for this is springs. For categorical problems that deal with the probability directly, a different conceptual model of forces and energies is needed to help visualize the fit.

The gas-pressure cylinder works perfectly. The piston is at some distance p from the bottom of the cylinder. At $p=1$, we make the pressure inside and out the same, so the force acting on the piston is zero. As the piston is moved down the cylinder, the force increases as the reciprocal of the distance (Boyle's law). If we integrate this from 1 to p , the energy stored in the cylinder becomes $-\log(p)$.

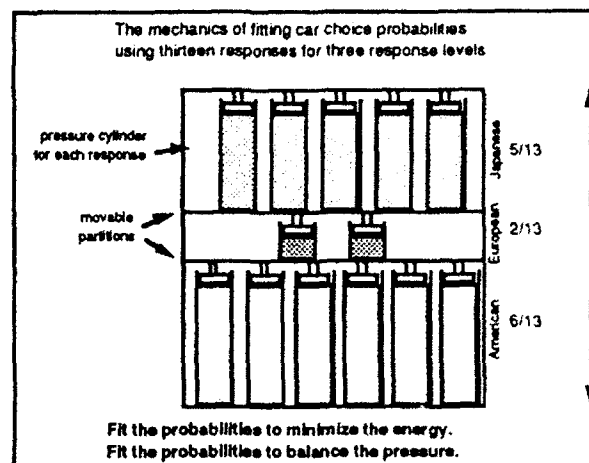
Give every response a pressure cylinder.



Pressure = $1/\text{height}$
Energy = $-\log(\text{height})$
Pressure(1) = 0
Pressure(0) = ∞

Improbable Events cost "energy"

To see how the machine works for fitting probabilities consider estimating the probability that a person will buy an American, European, or Japanese car. Let's take 13 observations and suppose that we obtain 6 cars that were American, 2 European, and 5 Japanese. For each type of car, place a gas cylinder in the compartment for that type of car in the frame shown below. The frame compresses the cylinders so that the total distance is one. The movable dividers between compartments are adjusted so that the energy in the gas cylinders is minimized, which also when the forces on each side of the compartment dividers is balanced. The result is the familiar estimates the probabilities, $p_i = n_i/n$.



$$\frac{n_J}{p_J} = \frac{n_E}{p_E} = \frac{n_A}{p_A} \quad \text{make the forces equal}$$

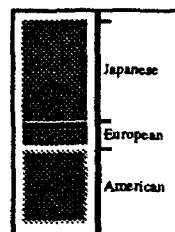
$$p_J + p_E + p_A = 1 \quad \text{make them sum to 1}$$

The solution:

$$p_J = \frac{n_J}{n_J + n_E + n_A}, \quad p_E = \frac{n_E}{n_J + n_E + n_A}, \quad p_A = \frac{n_A}{n_J + n_E + n_A}$$

To put this idea into a graph, consider the divided bar chart. Now imagine gas pressure in each compartment pushing on the dividers separating the compartments.

Divided Bar Chart



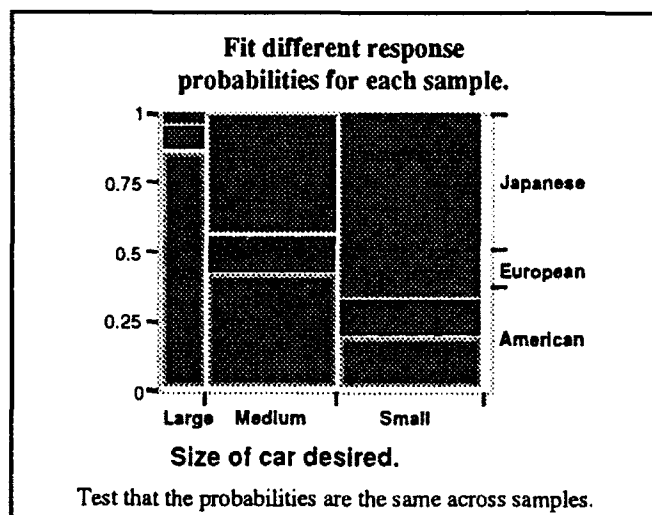
Shows the partition for each level such that the sum is 1.

Consistent with the pressure cylinder vision.

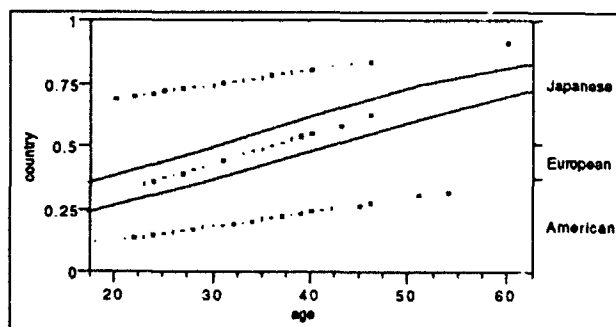
"Mosaic Plot"

To test that the probabilities are some hypothesized value, constrain the fit and measure how much additional energy was needed to impose the constraint.

In the one-way categorical layout, there is a set of compartments for each group. It is natural to test that the probabilities are the same across groups, by so constraining the fit, and adding up the change in energy for a likelihood ratio test.



If you have a continuous predictor for a categorical response, a similar idea works. The partition of the response probabilities is now a curve subject to the parameters in a logistic model. The curve balances forces from pressure cylinders placed horizontally at the predictor value, and in the partition corresponding to the observed response. Outliers are those points that did occur, but with probability squeezed down to a small value.



Conclusion

Statistical visualization needs conceptual models to relate what we see to statistical concepts. Think *mechanics* rather than *geography*. Think in terms of balancing of forces and the minimization of energy. Then you will understand what the statistics are really doing rather than just decorating the expression of quantities.

Leverage plots, comparison circles, and mosaic plots are three tools that harness the power of the conceptual models to serve our understanding of the fit.

References

- Belsley, Kuh, and Welsch (1980), *Regression Diagnostics*, New York, John Wiley & Sons.
- Cook, R.D., Weisberg, S., (1982), *Residuals and Influence in Regression*, New York: Chapman and Hall.
- Farebrother, R.W. (1987) "Mechanical Representation of the L1 and L2 Estimation Problems", ed. Y. Dodge, *Statistical Data Analysis Based on the L1 Norm and Related Methods*, Amsterdam: North-Holland Publishing Company, 455-464.
- Hartigan J.A. and B. Kleiner (1981), "Mosaics for contingency tables", *Proceedings of the 13th Symposium on the Interface between Computer Science and Statistics* W. F. Eddy editor, New York: Springer.
- Norman, Don (1989), *The Design of Everyday Things*.
- Sall, John (1991), "The Conceptual Model Behind the Picture", *ASA Statistical Computing and Statistical Graphics Newsletter* 2:1 5-8.
- Sall, John (1991), "The Conceptual Model for Categorical Responses", *ASA Statistical Computing and Statistical Graphics Newsletter* 2:2 33-36.
- Sall, John (1992), "Graphical Comparison of Means", *ASA Statistical Computing and Statistical Graphics Newsletter* 3:1 27-32.
- Sall, John (1990), "Leverage Plots for General Linear Hypotheses," *American Statistician*, 308-315.

ViSta: A Visual Statistics Research and Development Testbed

Forrest W. Young, Richard A. Faldowski & Mary M. McFarlane

Psychometric Laboratory, University of North Carolina
CB 3270 Davle Hall, Chapel Hill NC 27599-3270
uluru@unc.bitnet 919-962-5038

Abstract: We describe ViSta, a research and development testbed for statistical visualization techniques. Our work focuses on visualization techniques (1) for entire data analysis sessions; (2) for high-dimensional multivariate data; (3) for exploring models of high-dimensional data; and (4) for guiding novice data analysts.

1.0 Design Goals and Principles

ViSta is a Visual Statistics research and development testbed for visualization techniques designed to improve the quality, accuracy and satisfaction of the statistical data analysis process. Included are techniques for visualizing the overall structure of the data analysis session; for visualizing the results of analyses; for visually exploring the effects of reparameterizing models; and for visually guiding novice analysts.

ViSta is designed for an audience of users having a very wide range of data analysis sophistication, ranging from novices to experts. ViSta provides data analysis environments specifically tailored to the user's level of expertise. Guidance is available for novices, and tools are available for experts to create guidance for novices. A structured graphical user interface -- called a workmap -- is available for competent users, and a command line interface is available for sophisticated users.

ViSta's design understands that visualization techniques are not useful for everyone all of the time, regardless of their sophistication. Thus, all visualization techniques are optional, and can be dispensed with or reinstated at any time. In addition, standard non-visual data analysis methods are

available, including a command-line interface and support for batch jobs. ViSta also provides familiar textual reports of analysis results. So, while it may be that a single picture is worth a thousand numbers, this is not true for everyone all of the time, and, in any case, a picture *and* a table are worth more than either by itself.

ViSta combines its novel visualization techniques with standard statistical system features that have proven useful over the years. This combination means that ViSta provides a visual environment for doing data analysis without sacrificing the strengths of command lines, batch processing, and textual reports. ViSta's design rests on the assumption that combining traditional approaches with cutting edge visualization techniques gives the user the most complete understanding of the data.

These design principles are reflected throughout our discussion of the ViSta environment (Section 3.0) and of ViSta statistical visualization techniques (Section 4.0). ViSta is written in Lisp using the Lisp-Stat environment for statistical and dynamic graphics developed by Tierney (1991). Contact the first author about availability.

2.0 Visualization Techniques

ViSta emphasizes four statistical visualization techniques. These techniques are outlined here, and are discussed in greater detail in the remainder of this paper.

Session Visualization - The overall structure of the on-going data analysis session is presented visually as a "workmap" of the session. As the session progresses, the workmap grows

showing each step taken. Thus, when data are input, an icon appears that represents the data. When an analysis takes place, a "method" icon appears that is connected to the data icon, representing the fact that the data have been processed by the method. Then, a "model" icon appears connected to the method icon. This shows that the data have flowed through the method to yield a model. This workmap can be used to remind the analyst about the analysis that has taken place, and can be used to return to previous analysis steps. Workmaps are based on work of Young & Smith (1991).

Statistical Visualization - Statistical visualization techniques are designed to accurately and efficiently communicate data structure and modeling results. Results are presented visually via dynamic statistical graphics. Included are scatterplot matrices, scatterplots, spin-plots, tour-plots, histograms, box-plots, etc. These graphics are displayed in multiple windows that are linked by their observations and their variables. When an observation is high-lighted or labeled in one window, it can be high-lighted or labeled in any of the other windows. These graphics are designed to help the analyst visually explore spaces representing the data's structure or the structure of a model of the data. The graphics are complemented by standard textual reports. The concepts here are based on work by Stuetzle (1987) & Young, Faldowski & Harris (1992).

Interactive Graphical Modeling - Interactive graphical modeling techniques are designed to help the analyst visually explore the effects of revising a model's parameter estimates. The parameter estimates are represented by graphical elements such as points or vectors. These estimates can be revised with graphical tools for moving points or vectors. Once new estimates are obtained, the implications of the new estimates are rapidly displayed as changes in the model, its fit, and its residuals. While the new estimates may not be mathematically optimal, they may be more meaningful, leading to greater insight about the data. Geometrically, views of the data and its model are simultaneously manipulated in high-dimensional space, giving new views of the data and model which are unchanged relative to each other. These concepts are discussed further by Young, Faldowski & Harris (1992), Faldowski (1992), and McFarlane (1992).

Guidance Visualization - Users with no knowledge about data analysis can benefit from an environment which visually guides their analysis. To this end, the sequence of steps which expert data analysts think should be taken are presented visually as a cyclic graph. The graph guides those with less expertise through the series of steps in a complete statistical data analysis. The concepts presented here are based on work by Lubinsky, Young & Frigge (1990).

3.0 The visual data analysis environment

One of ViSta's main design principles is that a data analysis environment should reflect the sophistication of the user's data analysis knowledge. Thus, one goal of ViSta's software design is to maximize the data analyst's productivity and satisfaction by providing the analyst with a suitable data analysis environment.

ViSta is designed to accommodate the complete range of sophistication on the part of data analysts, from novice to expert. Since the data analysis environment which does this for a novice user is different from the one which does this for a sophisticated analyst, the design includes four kinds of environments. These environments include:

1. **Guidemaps** to guide novice data analysts through complete data analyses.
2. **Workmaps** to inform competent data analysts of the overall structure of their data analysis sessions.
3. **Command Line Interface** to let sophisticated data analysts dispense with the visual aids when they find them unnecessary.
4. **Guidance Tools** to let expert data analysts create guidance diagrams that are used by less expert analysts.

These four environments are very tightly coupled -- seamlessly integrated -- within ViSta. Analysts can switch between them whenever desired. We discuss each of the environments in this section, along with the notion of tight coupling. Finally, in addition to these four environments which are all highly interactive, ViSta provides a batch mode for automated analyses in repetitive data analysis situations. This mode is also discussed in this section.

3.1 Guidemaps for Novice Users

For novice data analysts the ViSta environment guides the user through the data analysis. Guidance is provided by a visual diagram that indicates to the novice analyst which steps should be chosen next - a *guidemap*. The structure of the guidemap doesn't change as the analysis proceeds although its highlighting changes. The steps are indicated by buttons, and the sequence of steps by arrows pointing from one button to the next. Figure 1 shows an example of a guidemap for multiple regression.

The user makes choices by pointing and clicking on the buttons with a mouse. Active buttons (which are dark) are suggested actions, whereas inactive buttons (the light ones) are actions that are not suggested. After a suggested action is taken the selection of active buttons changes to show

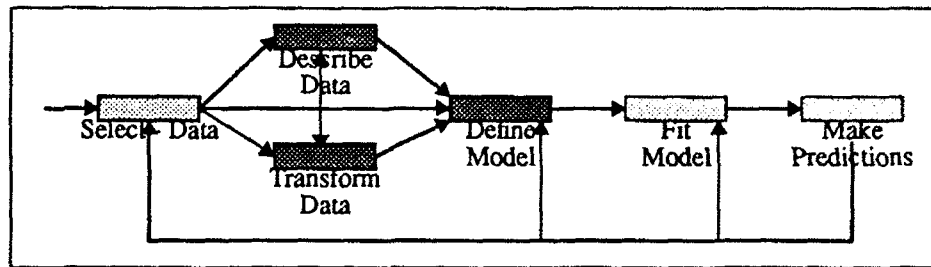


Figure 1 - A Guidemap

user which actions can be taken next. In this diagram the user has already selected data - that button is inactive and the following buttons are active. When the user clicks on one of the three active buttons, the corresponding action takes place, the button lightens in color, and the following buttons become active. For example, once the model is defined by clicking the "Define Model" button the "fit-model" button becomes inactive. Note that the guidemap is a cyclic graph whose nodes are possible actions, and whose edges are the possible action sequence. For more detail see Lubinsky, Young & Frigge (1990).

3.2 Workmaps for Competent Users

ViSta provides a graphical interface for competent data analysts called a *workmap*. A workmap is a visual diagram of the steps taken in the analysis. Unlike a guidemap, which doesn't change, a workmap is created and expands as the analysis takes place. The user points and clicks to perform

analyses and to create the structured analysis diagram. Note that the workmap is an acyclic graph whose nodes are possible actions and whose edges are the sequence of possible actions. For more details see Young & Smith, 1991.

The workmap shown in Figure 2 is a structured analysis diagram created during a ViSta analysis session. In this analysis the analyst first loaded a datafile named "car-ratings", creating a data icon with the same name. These data were then standardized, creating a new data object with an icon named "STD:car-ratings". The analyst then loaded a second data named "car-pref14", creating a third data object and another data icon with the same name. These data were analyzed using the "PrinComp" method for principal components analysis. This produces a method icon named "PrinComp", and a model icon named "PCA:car-pref14". The analyst then requested that the model create three new data objects: scores, coefficients and input data. Finally, the analyst merged the standardized ratings with the principal components

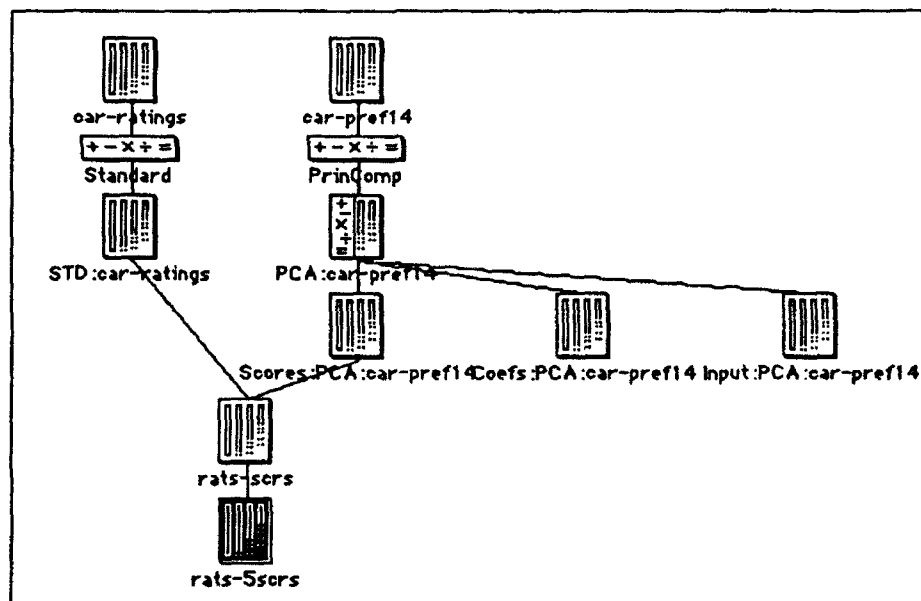


Figure 2 - A Workmap


```

> (def car-ratings (load-data ":ViSta:Data:car-ratings.lsp"))
CAR-RATINGS
> (def std-car-ratings (standardize-data :dialog nil :mean 0 :stdv 1))
STD-CAR-RATINGS
> (send std-car-ratings :report-data)
NIL
> (def car-prefs (load-data ":ViSta:Data:car-pref14.lsp"))
CAR-PREFS
> (def pca-car-prefs (principal-components :dialog nil :corr t))
PCA-CAR-PREFS
> (send pca-car-prefs :create-data-objects :dialog nil
  :scores t :coefs t :input t)
*Object: 1870806, prototype = MV-DATA-OBJECT-PROTO
>

```

Figure 3 - Command Line Interface

nent scores. Any of the icons in this diagram can be opened in various ways to visualize or report data or results. This example corresponds to the example in Section 3.3 on command lines, and the example in Section 3.5 on batch mode.

3.3 Command Lines for Sophisticated Users

For sophisticated data analysts, ViSta provides a command line interface. These commands, which are shown in Figure 3, are entered through the keyboard, causing the analysis to take place. They also create the workmap (which may be hidden, if desired). The commands in Figure 3 follow the same series of steps as those outlined in Section 3.2, and generate the same workmap shown there.

3.4 Guidance Tools for Expert Users

For expert analysts ViSta is designed to provide tools to create guidance diagrams that can be used by other users. These diagrams can be constructed by using the mouse to point and click, or by using the command line to type commands. A guidance diagram has already been shown in Section 3.1¹.

3.5 Batch Mode - Automated Analysis in Repetitive Situations

The four kinds of environments discussed above are all *highly interactive*. This means that as soon as an icon is clicked, or a command is typed, ViSta responds. This is desirable in many situations, especially when analyses are being performed on a one-shot or exploratory basis. However, in other situations, such as when an analysis will be repeated again in the future on a new wave of data, it is preferable to be able to collect all commands together into a file and run them all at once without user interaction. This is

called "batch" mode because all commands are run as batch.

An example of a ViSta batch mode file is shown in Figure 4

```

(def car-ratings (load-data "car-ratings.lsp"))
(def std-car-ratings
  (standardize-data
   :dialog nil
   :mean 0
   :stdv 1))
(send std-car-ratings :report-data)
(send std-car-ratings :visualize-data)
(send std-car-ratings :summarize-data)
(def car-prefs (load-data "car-pref14.lsp"))
(def pca-car-prefs
  (principal-components
   :dialog nil
   :corr t))
(send pca-car-prefs :report-model)
(send pca-car-prefs :visualize-model)
(send pca-car-prefs :create-data-objects
  :dialog nil
  :scores t
  :coefs t
  :input t)

```

Figure 4 - Batch Mode

In this example the analyst has loaded data concerning car ratings, which are standardized. He/she then obtains a report (listing) of these data, followed by a visualization and some summary statistics. The analyst then loads data about car preferences. These data are submitted to a principal components analysis. A report and a visualization is obtained of the results and then output data objects are created. This batch code corresponds to the analyses discussed in Section 3.2 on workmaps and in Section 3.3 on the command line interface.

3.6 Tight Coupling of All Environments

The five data analysis environments are tightly coupled, as can be seen from the previous sections. The guidance diagrams used by novice analysts generate commands that are identical to those typed by sophisticated users with the command line interface. The graphical interface used by competent analysts also generates the same commands. The

1. Note that guidemaps and guidance tools are not yet implemented at the time this paper is being written, but we anticipate they will be by the time it is published.

commands, in turn, generate the structured analysis diagram and perform the data analysis. These commands can be used in batch files.

It is possible to switch between the several kinds of environments at any time. When the sophisticated user moves into an unfamiliar type of data analysis, or when any analyst loses track of the overall structure of the analysis, the analyst can switch from the command line interface to the graphical interface, with the entire structured history of the analysis session being presented. Similarly, the moderately competent analyst can switch guidance diagrams on or off as desired.

4.0 Statistical Visualization Methods

One of the main design principles of ViSta's Statistical Visualization methods is that a single picture is worth a table of a thousand numbers. Statistical Visualization uses geometrically based statistical models to provide visual insight into the structure of data. Consider the general linear model. It can be viewed as a geometric model that represents observations as points in one or more high-dimensional spaces. These spaces have dimensions for variables or for linear combinations of variables. The statistical visualization of a general linear model presents the results of the analysis as a group of interacting plots, the purpose being to intuitively communicate the results of the analysis through pictures. When this visualization is combined with traditional reporting techniques (i.e., tables of results), the user gains a greater understanding of the results than when either technique is used alone. That is, most of the time most of us find a picture and a table to be worth more than either by itself.

Three kinds of statistical visualization tools are available in ViSta. They are used for three different purposes:

1. **Linked Plots:** This set of statistical visualization tools is used to present data structure and to present the results of statistical analyses.
2. **Guided Tours and Spreadplots:** These statistical visualization tools are used to explore the structure of high-dimensional data and of models of such data.
3. **Interactive Graphical Modeling:** This set of statistical visualization tools is used to help search for meaningful and parsimonious model parameterizations.

These statistical visualization aspects of ViSta are discussed next, followed by a discussion of ViSta's more traditional reporting techniques.

4.1 Empirically Linked plots - Groups of Interacting Plots

One of the primary statistical visualization tools in ViSta is the linking of several plots through their data's observations and variables. This was discussed by Stuetzle (1987). We call these "empirically" linked plots since they are linked through the data. In his paper he discussed linking via the observations of the data. In ViSta, plots can be empirically linked via either the data's observations, or their variables.

Figure 5 presents an example of a layout of four plots. At the upper-left is a plot-matrix. To its right is a spin-plot. At the bottom-left is a scatterplot, and to its right is a histogram. These plots are empirically linked via their observations. The labeled points in the scatterplot are highlighted in the spin-plot and histogram (and could be in the plot-matrix). As one selects points in any plot (by clicking or dragging the mouse), points for the same observations can be highlighted in any other plot. Being able to see where observations appear in several plots lets the analyst get a better idea of the data's structure.

The four plots are also empirically linked via their variables. By clicking on a cell of the spreadplot the user can choose which variables are plotted in the other plots. Notice that the scatterplot and histogram are showing variables which correspond to the cell in the plot-matrix which has the "finger cursor" located on it. These are also two of the spin-plot's three variables. Clicks and shift-clicks on cells in the scatterplot-matrix determine which variables appear in the other plots. Being able to display various combinations of variables in the other plots lets the analyst look at many views of the data's structure.

4.2 Guided Tours - High-Dimensional Spinnable Plots

A tour-plot (the large window in Figure 6) is a spin-plot that spins in more than 3 dimensions (Asimov, 1985; Buja & Asimov, 1986; Young, Kent & Kuhfeld, 1988). A guided tour plot spins as directed by the user. Just as a spin-plot is designed to help the user understand structure in three dimensional data, a guided tour-plot is designed to help the user understand structure in high-dimensional data.

ViSta's implementation of a guided tour is shown in Figure 6. This implementation uses real-time dynamic graphics which are guided by the user with high-interaction, immediate feedback, point-and-click mouse actions. ViSta's guided tour lets the user create and control rotation in a portion of high-dimensional space which can have up to six dimensions.

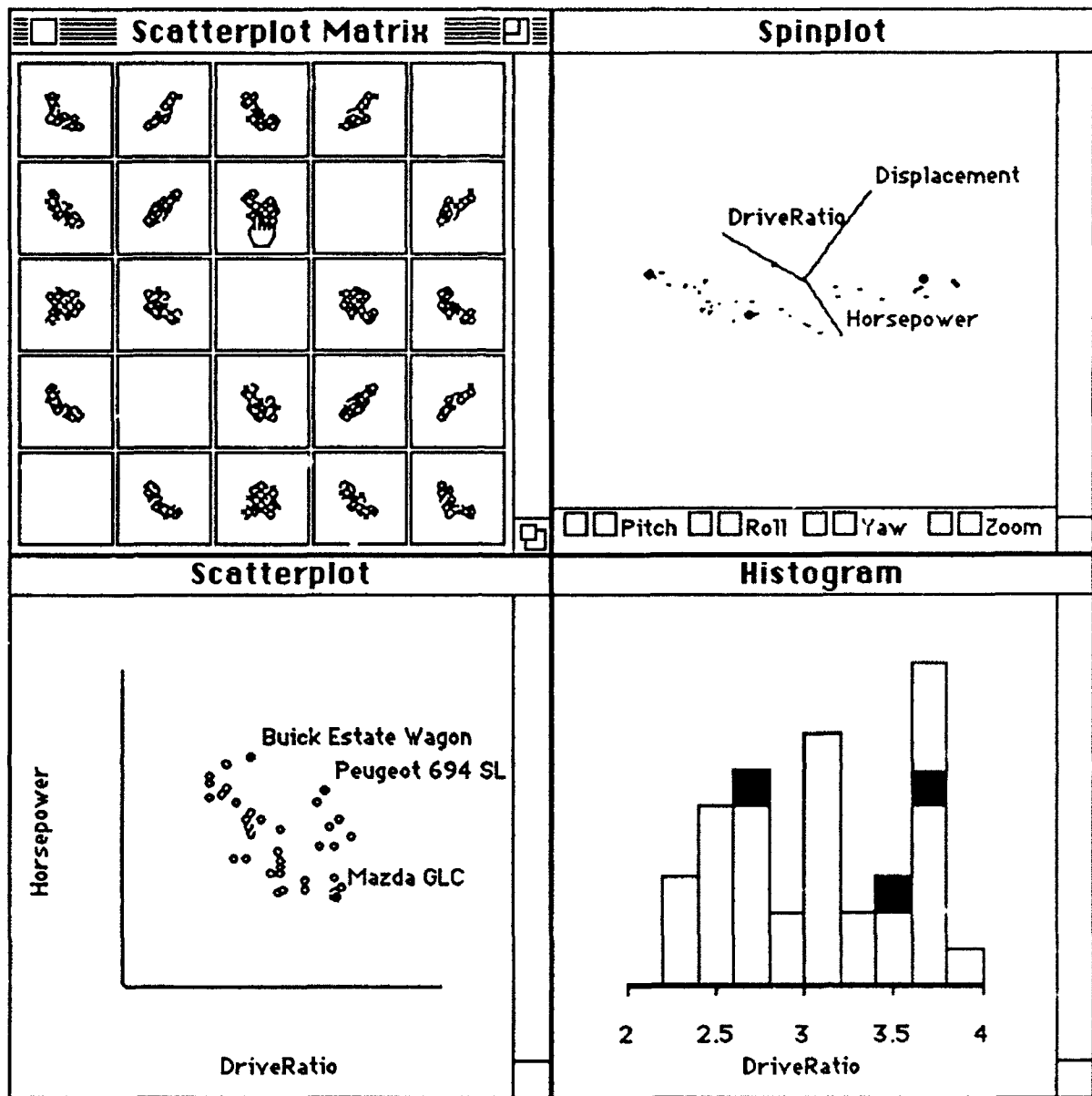


Figure 5 - Empirically Linked Plots

sions. The user can also control which six-dimensional portion of the high-dimensional space is used for rotation.

The Guided Tour figure shows a group of four linked plots. In addition to the tour-plot in the large window, there are two spin-plots ("Left Target" and "Right Target"), and a partially hidden scatterplot matrix. High-dimensional spinning (touring) takes place from the Left Target's orientation to the Right Target's orientation. The scatterplot matrix is used to determine which variables appear in the two target windows.

The figure shows the guided tour prior to spinning, so it shows a point cloud that is the same as the Left Target. When the space has spun 90°, the cloud in the guided tour window will be the same as the Right Target. When it spins a further 270° it will return to the Left Target orientation. Spinning can occur horizontally (involving the two targets' horizontal axes), vertically, or both at once. In addition, the third axis in each target can also be involved in spinning, so that spinning can involve all six axes. See Young & Rheingans (1991b) for a video example of guided tours.

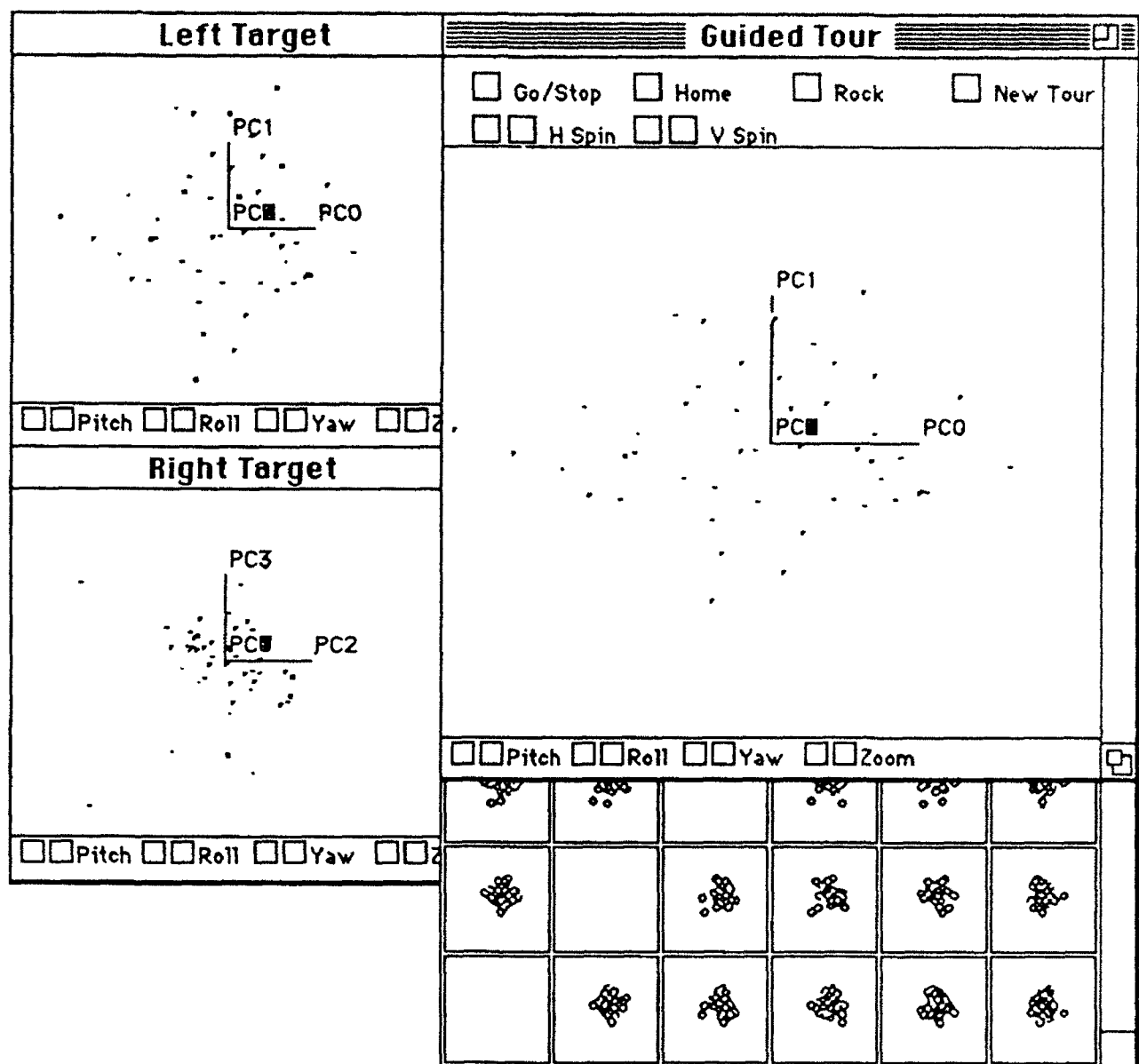


Figure 6 - A Guided Tour Spreadplot

4.3 Spreadplots - Algebraically Linked Plots

ViSta includes spreadplots (Young, Faldowski & Harris, 1991) among its statistical visualization techniques. A spreadplot is the graphical equivalent of a spreadsheet: It is a group of several interacting dynamic plots, with the several plots being *algebraically linked by equations*. Note that algebraic linkage is fundamentally different from empirical linkage. Empirical linkage involves the data's observations and variables. Algebraic linkage involves a model's equations. ViSta's spreadplots can have both kinds of linkages between plots in the same spreadplot.

Figure 6 is titled a "Guided Tour Spreadplot" because there are algebraic links between the plots. There are two kinds of algebraic links. First, there are equations which link the two target plots with the tour-plot. These equations create the specifics of the high-dimensional spinning that occurs in the tour-plot. The user actually has the choice of two sets of equations for two different types of tours. One set of equations (Buja & Asimov, 1986) implements the high-dimensional rotation model mentioned above. The other set (Young, Kent & Kuhfeld, 1988) implements a linear interpolation model. Both are discussed by Young & Rheingard (1991a).

The second type of algebraic link between the group of plots implements the residualization model proposed by Young, Kent & Kuhfeld (1988). When the "New Tour" button is clicked the specific position of the tour-plot in its spin between the two targets is used, along with the residualization equations, to update the two target windows. These new targets are then used, via the high-dimensional spinning equations that link the windows, to modify the path taken by the tour-plot during its high-dimensional spin.

4.4 Interactive Graphical Modeling - Graphical Tools for Fitting Models

Interactive graphical modeling is the final statistical visualization technique in ViSta. An example is shown in Figure 7. This is a statistical visualization technique for visually exploring the nature of alternative parameterizations of statistical models. The technique uses graphical tools to modify a model's parameterization, with the implications of the modifications being displayed as changes in the dynamic graphs that portray the model, its residuals, and its fit. A data analyst would use this tool to explore for a model of the data

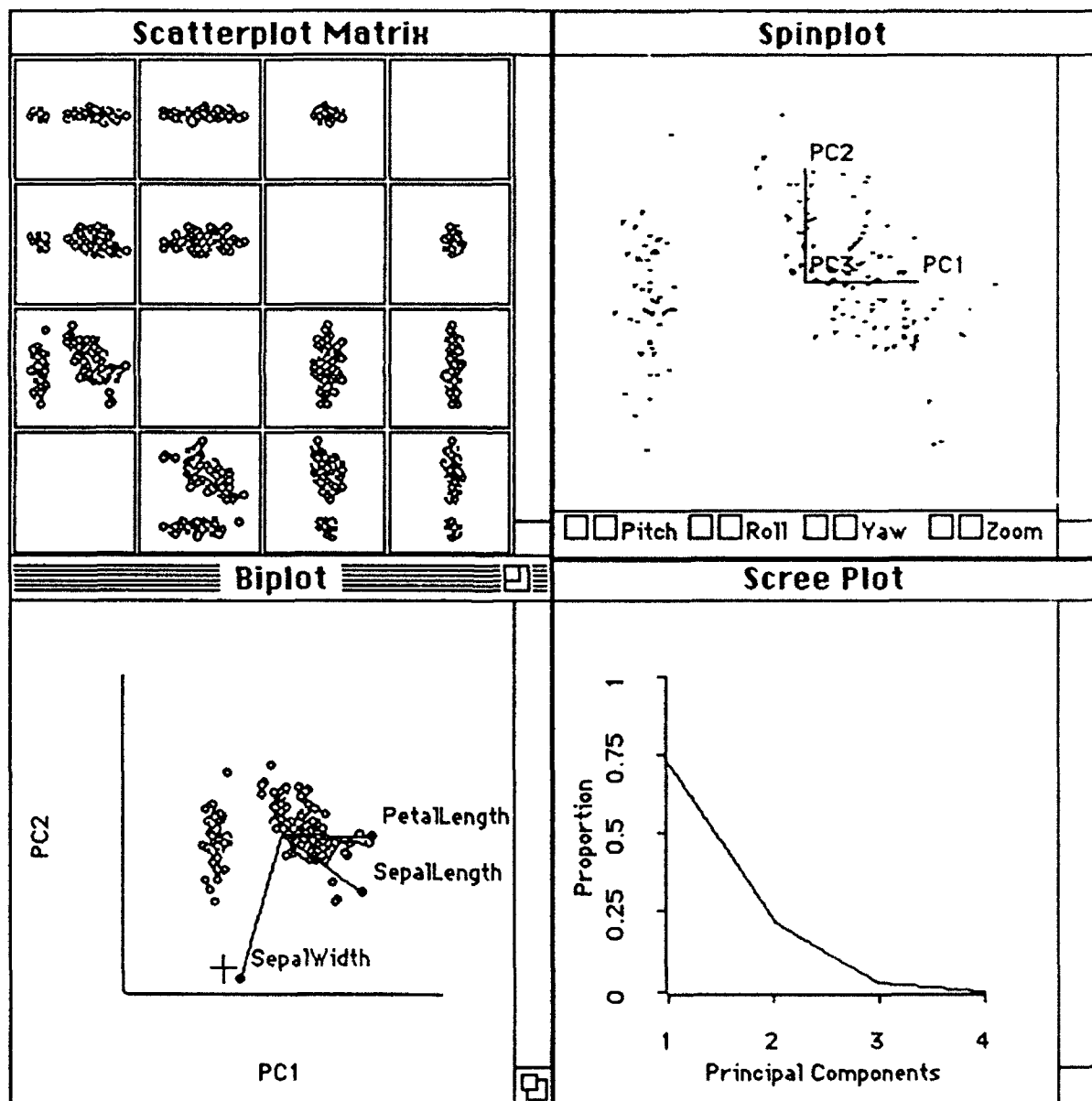


Figure 7 - Interactive Graphical Modeling

which provides better understanding of the data than the one provided by a traditional algebraic analysis.

Interactive graphical modeling assumes that an explicit data analysis model has been fit to the data by some means that generates initial estimates of the model's parameters. Currently, research and development is underway for interactive graphical modeling that are appropriate for multiple regression, principal components and multidimensional scaling, all of which are fit, in ViSta, by standard OLS techniques.

Figure 7 presents the results of fitting the Principal Components model to the well known Fisher Iris data. At the top are a scatterplot matrix and a spin-plot of the scores of the irises on the principal components. The lower left-hand "Biplot" (Gabriel, 1981) is a scatterplot of these scores plus a vector plot of the coefficients that the variables have on the principal components (variables are labeled, one being hidden). These coefficients are the model's parameters. The "Scree Plot" is a plot of the proportion of variance accounted for by each component.

The user can interactively change the model of the data by using the graphical "vector-moving" tool that is available in the biplot. This tool is represented by the cross-shaped cursor shown located near the tip of the "Sepal Width" vector in the biplot. When this tool is located directly on one of the vector tips, the vector tip can be dragged to a new location by holding the mouse button down and moving the mouse. This creates new estimates of the model's parameters, which in turn cause new scores and fit to be calculated, which in turn are used to modify all four plots. Note that the set of plots are a spreadplot, since they are algebraically linked via the equations that re-estimate the model and its fit when parameters are modified.

4.5 Reports - Traditional Reporting Techniques

The visualization techniques that have been outlined above are state-of-the-art features for reporting data analysis results. However, our design sees them as supplementing, not supplanting, the traditional methods for reporting results that have been developed and widely used over the past several decades. Thus, ViSta is designed to have a full range of standard reporting techniques. For example, when a Multivariate Regression analysis is performed, the user can view the report shown in Figure 8. Similarly, when simple statistics are desired about data, the user can obtain the report shown in Figure 9. Thus, when these traditional reporting methods are combined with ViSta's statistical visualization reports, the data analyst has the greatest information for understanding data that current technology can provide.

MULTIVARIATE REGRESSION ANALYSIS			
Dependent Variables: (Chinups Situps)			
Independent Variables: (Weight Waist)			
For Dependent Variable Chinups			
Least Squares Estimates:			
Constant	46.4511	(12.624)	
Weight	0.0801815	(0.0857006)	
Waist	-1.44976	(0.66084)	
R Squared:	0.338996		
Sigma hat:	4.54364		
Number of cases:	20		
Degrees of freedom:	17		
For Dependent Variable Situps			
Least Squares Estimates:			
Constant	634.437	(137.982)	
Weight	0.717825	(0.93672)	
Waist	-17.4319	(7.22307)	
R Squared:	0.43627		
Sigma hat:	49.6627		
Number of cases:	20		
Degrees of freedom:	17		
Figure 8 - Regression Report			

5.0 Conclusion

In this paper we have presented ViSta, a testbed for research and development in statistical visualization. Our work emphasizes visualization techniques (1) for entire data analysis sessions, (2) for high-dimensional multivariate data; (3) for exploring models of high-dimensional data; and (4) for guiding novice data analysts.

We believe that data analysis systems of the 21st century will incorporate methods like those we have presented, and that they will help the data analyst to have a more insightful, productive and satisfying data analysis experience than is possible in current data analysis environments.

DESCRIPTIVE STATISTICS FOR cars					
VARIABLE	MEAN	STDV	VARIANCE		
MPG	24.76	6.55	42.87		
Weight	2.86	0.71	0.50		
DriveRatio	3.09	0.52	0.27		
Horsepower	101.74	26.44	699.33		
Displacement	177.29	88.88	7899.08		
MINIMUM, 2nd QUARTILE, MEDIAN, 4th QUARTILE, MAXIMUM					
MPG	15.50	18.55	24.25	30.25	37.30
Weight	1.92	2.21	2.69	3.41	4.36
DriveRatio	2.26	2.70	3.08	3.61	3.90
Horsepower	65.00	79.00	100.00	122.50	153.00
Displacement	85.00	105.00	148.50	228.00	360.00
INTERQUARTILE RANGE, RANGE					
MPG	11.70	21.80			
Weight	1.20	2.45			
DriveRatio	0.91	1.64			
Horsepower	43.50	90.00			
Displacement	123.00	275.00			
CORRELATION MATRIX					
1	-0.9	0.42	-0.87	-0.79	MPG
-0.9	1	-0.69	0.92	0.95	Weight
0.42	-0.69	1	-0.59	-0.8	DriveRatio
-0.87	0.92	-0.59	1	0.87	Horsepower
-0.79	0.95	-0.8	0.87	1	Displacement

Figure 9 - Simple Descriptive Statistics

6.0 References

- Asimov, D. (1985). "The Grand Tour: A Tool for Viewing Multidimensional Data," *SIAM J. Scientific and Statistical Computing*, 6, 128-143.
- Buja, A. and Asimov, D. (1986). "Grand Tour Methods: An Outline," *Computer Science and Statistics: Proc. 17th Symposium on the Interface*, Elsevier, Amsterdam.
- Faldowski, R.A. (1992). *Interactive Graphical Modeling*. Ph.D. Dissertation Proposal, Univ. N. Carolina Psychometrics Laboratory, Chapel Hill, NC.
- Gabriel, K. R. (1981). "Biplot Display of Multivariate Matrices for Inspection of Data and Diagnosis," in V. Barnett (ed.): *Interpreting Multivariate Data*. Wiley, London.
- Lubinsky, D.J., Young, F.W. & Frigge, M.L. (1990). *Representing and Using Data Analysis Strategies*. Technical Report, Bell Telephone Laboratories, Holmdel NJ.
- McFarlane, M.M. (1992). "Interactive Graphical Modeling for Multidimensional Scaling. Unpublished Master's Thesis, UNC Psychometrics Laboratory, Chapel Hill NC.
- Stuetzle, W. (1987). "Plot Windows," *J. American Statistical Association*, 82, 466-475.
- Tierney, L. (1991). *Lisp-Stat: An Object-Oriented Environment for Statistical Computing and Dynamic Graphics*. Wiley, New York.
- Young, F.W., Faldowski, R.A. & Harris, D.F. (1992). *The Spreadplot: A graphical spreadsheet of Algebraically Linked Dynamic Plots*. *ASA Proceedings of the Section on Statistical Graphics*, (in press)
- Young, F. W., Kent, D. P. and Kuhfeld, W. F. (1988). "Dynamic Graphics for Exploring Multivariate Data," in Cleveland, W. S. and McGill, M. E. (eds.): *Dynamic Graphics for Statistics*. Wadsworth, Inc., Belmont, Calif
- Young, F.W. & Rheingans, P. (1991a). *Visualizing Structure in High-Dimensional Data*. *IBM Journal of Research and Development*. 35, 97-107.
- Young, F.W. & Rheingans, P. (1991b). *Visualizing Multivariate Data with VISUALS/Pxpl. (Video)*. *IBM Journal of Research and Development*. 35, (video supplement).
- Young, F.W. & Smith, J.B. (1991). *Towards a Structured Data Analysis Environment: A Cognition-Based Design*. In: W., Buja, A., & Turkey, P. (Eds.) *Computing and Graphics in Statistics. IMA Volumes in Mathematics and its Applications*, 36, Springer Verlag. 252-279.

ADVANTAGES AND DISADVANTAGES OF
DENSITY ESTIMATION WITH WAVELETSGilbert G. Walter¹
and
Jugal K. GhoraiDepartment of Mathematical Sciences
University of Wisconsin-Milwaukee
Milwaukee, WI 53201**Abstract**

Wavelets constitute a new orthogonal system which has many practical and theoretical advantages over the classical systems. It gives better localization properties as well as better convergence properties. This gives it better asymptotic properties when used to construct estimators of a probability density function. However, its small sample properties seem to have little advantage over the kernel methods and indeed often do not give as close an approximation to the true density.

1. Introduction.

The subject of nonparametric probability density estimation has spawned a huge (mostly forgettable) literature. In addition to the traditional histogram method there are methods based on kernels (Parzen (1962), Rosenblatt (1956)), on Fourier series (Kronmal and Tarter (1968)), on Fourier transforms (Davis (1974)), on orthogonal polynomials (Schwartz (1967)), on splines (Wahba (1975)) and on general delta sequences (Földes and Revez (1974)). Most of the methods give mean square and almost sure consistent estimators at points of continuity of the density, but most converge slowly even when the density is infinitely differentiable. Some, the higher order methods, have more rapid rate of convergence for such densities but may fail to converge for densities that are merely continuous (Walter and Blum (1979)).

With so many choices available it seems superfluous to introduce yet another method of density estimation. But the method to be discussed based on orthogonal wavelets (Daubechies (1988)) seems to have a number of advantages over other methods.

They are: (i) the estimators are at the same time consistent for continuous densities and rapidly convergent for C^∞ densities; (ii) their parameters have an easy interpretation in terms of the scale; (iii) all calculation are recursive; (iv) the error is given in terms of simple wavelet expansions; (v) their localization properties lessen the effect of outliers. However it is not clear that they will perform better as density estimators in relatively small samples. The principal thrust of this work will therefore be to compare estimators based on wavelets with those based on kernels. The procedure will be to generate samples from various densities, construct the estimators and see how they compare. We first present some background in wavelets, and some theoretical properties of the estimators.

2. Background on Wavelets.

The subject of wavelets has had an extremely rapid development in the last few years. It began with a technique, poorly understood first, that seemed to work for seismic analysis better than previous methods (Morlet et al (1982)). This involved representation of signal by means of integral transforms (Grossman and Morlet (1984)), but was soon extended to nonorthogonal series representations (Daubechies (1988), Heil and Walnut (1989)) and eventually to orthogonal series (Meyer (1988), Daubechies (1988)).

In addition to the applications in seismic analysis, wavelet representations have been found to be useful in image processing, signal analysis, and data compression (Beylkin et al (1989), Strang (1989), and Mallat (1989)). They have proved to be useful in pure mathematics also since they provide unconditional bases on certain Banach spaces such as H^1 and L^p (Daubechies (1988)).

The wavelet estimators we use in density estimation are based on orthogonal series but are similar to kernel

¹Research Supported by NSF Grant #DMS-901526

estimators. They may be given in the form

$$\hat{f}_m(x) = \sum_{n=-\infty}^{\infty} \frac{1}{N} \sum_{i=1}^N \phi_{m,n}(X_i) \phi_{m,n}(x), \quad x \in \mathcal{R}, m \in \mathcal{Z}. \quad (2.1)$$

where X_1, \dots, X_N is an i.i.d. sample and $\phi_{m,n}(x) = \gamma^{m/2} \phi(2^m x - n)$. Here $\phi(x)$ is the scaling function; its translates are orthogonal in $L^2(\mathcal{R})$ and $\{\phi_{m,n}(x)\}$ is an orthogonal basis of a subspace V_m of $L^2(\mathcal{R})$. This V_m also has a reproducing kernel $q_m(t, x)$ in terms of which (2.1) becomes

$$\hat{f}_m(x) = \frac{1}{N} \sum_{i=1}^N q_m(X_i, x) \quad x \in \mathcal{R}, m \in \mathcal{Z}. \quad (2.2)$$

A third expression is in terms of the wavelets $\{\psi_{m,n}\}$ themselves. These constitute an orthonormal basis of $L^2(\mathcal{R})$ and have the form $\psi_{m,n}(x) = 2^{m/n} \psi(2^m x - n)$ where $\psi(x)$, the "mother wavelet" is based on $\phi(x)$. The estimator is, for $m \in \mathcal{Z}$,

$$\hat{f}_m(x) = \sum_{k=-\infty}^m \sum_{n=-\infty}^{\infty} \left(\frac{1}{N} \sum_{i=1}^N \psi_{k,n}(X_i) \right) \psi_{k,n}(x), \quad x \in \mathcal{R}, \quad (2.3)$$

which is analogous to other orthogonal series estimators. The infinite series in (2.1) and (2.3) will be truncated in most cases since both $\phi(x)$ and $\psi(x)$ often have compact support. Even if they do not, the convergence is rapid and the tails may be safely ignored.

3. Examples of Wavelets.

There are three approaches to constructing orthogonal wavelets. They all involve construction of the scaling function $\phi(t)$ first.

In the case of Franklin Wavelets, the $\phi(t)$ is found by orthogonalization of the basic spline function $\theta(t) = \chi_{[0,2]}(t)(1 - |t-1|)$. That is, $\phi(t)$ is chosen in the space spanned by $\theta(t-n)$ in such a way that it is orthogonal to its translates. The resulting function is, in terms of its Fourier transform,

$$\hat{\phi}_F(\omega) = \frac{\sin^2(\omega/2)}{(\omega/2)^2} (1 - \frac{2}{3} \sin^2(\omega/2))^{-\frac{1}{2}}. \quad (3.1)$$

The scaling function for the Meyer Wavelets is also given in terms of its Fourier transform. It is chosen directly as a function whose Fourier transform satisfies the orthogonality condition and is given by

$$\hat{\phi}_M(\omega) = \left\{ \int_{\pi-\omega}^{\pi+\omega} dP \right\}^{\frac{1}{2}} \quad (3.2)$$

where P is a probability measure with support in $[-\frac{\pi}{3}, \frac{\pi}{3}]$. For this work we take $dP(\omega) = C((\frac{\pi}{3})^2 - \omega^2)^2 \chi_{\frac{\pi}{3}}(\omega)$ where $\chi_{\frac{\pi}{3}}$ is the indicator function of $[-\frac{\pi}{3}, \frac{\pi}{3}]$ and C is a normalizing constant, $(\frac{15}{16}(\frac{3}{\pi})^5)$. In the case of the Daubechies Wavelets, the scaling function is defined by its "dilation equation"

$$\phi_D(t) = \sum_n c_n \phi_D(2t - n) \quad (3.3)$$

where in the simplest case $c_0 = (1 + \sqrt{3})/4$, $c_1 = (3 + \sqrt{3})/4$, $c_2 = 1 - c_0$, $c_3 = 1 - c_1$ and $c_n = 0$ for all other values of n . These ϕ_D are continuous but not differentiable functions with support on $[0, 3]$.

By allowing more terms in (3.3) the $\{c_n\}$ can be chosen (Daubechies (1988, pg 980)) so that ϕ is smooth but has support on a larger interval.

4. Convergence Results.

As indicated previously, wavelets have nice convergence properties. These are based on the following lemma (Walter (1992)).

Lemma 4.1 Let the scaling function $\phi \in S_r$, $r \geq 0$, and suppose that

- (i) $\hat{\phi}(\omega) = 1 + O(|\omega|^\lambda)$
- (ii) $\sum_n \phi(t-n)e^{-i\omega(t-n)} = 1 + O(|\omega|^\lambda)$

as $|\omega| \rightarrow 0$ for some $\lambda \geq 1$; let $q_m(t, x)$ be the reproducing kernel of V_m . Then (i) $\{q_m(t, x)\}$ is a quasi-positive delta sequence on \mathcal{R} and (ii) the Sobolev norm $\|q_m(\cdot, y) - \delta(\cdot - y)\|_{-\alpha} = O(2^{-m\lambda})$ for $\alpha > \lambda + \frac{1}{2}$.

Remark 4.1 This hypothesis is satisfied for all scaling functions $\phi \in S_r$, $r \geq 0$ with $\lambda = 1$. For other ϕ , notably the Meyer Wavelet, λ may be taken arbitrarily large. These two properties are not shared by any of the classical orthogonal systems.

Theorem 4.1 Let the scaling function $\phi \in S_r$ and satisfy the hypothesis of the lemma for some $\lambda \geq 1$. Let X_1, \dots, X_N be an iid sample from the continuous bounded density $f(x)$. Define

$$\hat{f}_m(x) = \frac{1}{N} \sum_{i=1}^N q_m(x, X_i), \quad x \in \mathcal{R}. \quad (4.1)$$

Then

- (i) $E|\hat{f}_m(x) - f(x)|^2 \rightarrow 0$ uniformly on compact sets as $m \rightarrow \infty$ and $m = O(\log N)$,

- (ii) If $f \in H^\alpha$, $\alpha > \lambda + \frac{1}{2}$, $m \approx \frac{\log 2}{2\lambda+1} \log N$
 $E|\hat{f}_m(x) - f(x)|^2 = O(N^{-2\lambda/(2\lambda+1)}).$

Proof: The variance of the estimator is given by

$$\begin{aligned} E|\hat{f}_m(x) - f_m(x)|^2 &\leq \frac{1}{N} \int_{-\infty}^{\infty} q_m^2(x, y) f(y) dy \\ &\leq \frac{1}{N} \|f\|_{\infty} q_m(x, x) \\ &= \frac{1}{N} \|f\|_{\infty} 2^m q(2^m x, 2^m x) \\ &= O(2^m/N) \end{aligned} \quad (4.2)$$

since $q(x, x)$ is continuous and periodic. From this the results follow from the lemma and Schwartz inequality for the Sobolev spaces. \square

Wavelets do however, have one theoretical shortcoming in comparison to the classical orthogonal systems. The trigonometric series on $[-\pi, \pi]$, given by $\{e^{inx}\}$, has the property that the expansion of a periodic function $f(x)$ has partial sums

$$S_N(x) = \sum_{n=-N}^N a_n e^{inx} \quad (4.3)$$

which are invariant under translations. That is the N -partial sum of $f(x - \alpha)$ is $S_N(x - \alpha)$. This does not hold for most wavelets. The translates are not in the same subspace V_m .

5. Choice of m .

The asymptotic results are not of much use for choosing the parameter m . In general the coefficients in the density estimator are \hat{a}_{mk} and \hat{b}_{mk} where

$$\begin{aligned} \hat{a}_{mk} &= \frac{1}{N} \sum_{i=1}^N \phi_{mk}(X_i), \\ \hat{b}_{mk} &= \frac{1}{N} \sum_{i=1}^N \psi_{mk}(X_i). \end{aligned}$$

The dilation equation (3.3) and the relation

$$\psi(t) = \sum_k c_{k+1} (-1)^k \phi(2t + k) \quad (5.1)$$

give us the following decomposition algorithm.

$$\begin{aligned} a_{m-1,k} &= 2^{-\frac{1}{2}} \sum_j c_j a_{m,2k+j} \\ b_{m-1,k} &= 2^{-\frac{1}{2}} \sum_j c_{1-j} (-1)^j a_{m,2k+j}. \end{aligned} \quad (5.2)$$

We choose the finest scale \bar{m} of interest; calculate $\hat{a}_{\bar{m},k}$ from the data and then use (5.2) to find $\hat{a}_{\bar{m}-1,k}$ and $\hat{b}_{\bar{m}-1,k}$. We keep decreasing m until the $MISE$ takes a jump. It is not too difficult to show that

$$\begin{aligned} &MISE(m) - MISE(m-1) \\ &= \frac{1}{N} \int_{-\infty}^{\infty} 2^m q(2^m x, 2^m x) f(x) dx \\ &\quad - \frac{N+1}{N} \sum_k b_{m-1,k}^2. \end{aligned}$$

The integral above can be approximated by $\frac{1}{N} \sum_{i=1}^N q(2^m X_i, 2^m X_i)$. This will allow to choose an appropriate m .

6. Results of Simulation.

A sample of size 200 was taken from each of four different densities. These densities were formed by linear combinations of normal $f_N(x|\mu, \sigma)$ and double exponential densities $f_D(x)$ and were

- (i) $f_1(x) = f_D(x)$
- (ii) $f_2(x) = 0.7f_D(x) + 0.3f_N(x|-1.5, 0.6)$
- (iii) $f_3(x) = 0.3f_N(x|-1, 0.7) + 0.7f_N(x|0.5, 0.45)$
- (iv) $f_4(x) = 0.7f_D(x) + 0.3f_N(x|-1.5, 0.6)$.

The last two densities, f_3 and f_4 were in C^∞ but are not symmetric. The density f_3 is unimodal and f_4 is bimodal. This tested the ability of the estimators to discriminate between them. Most estimators should work well for these densities. In the case of f_1 , the density is not differentiable at $x = 0$; we were interested in determining which estimators would give a good approximation to the peak. In f_2 we again have bimodal density in which one peak is sharp and the other smooth.

Kernel estimators were based on two different kernels

$$(i) \quad k_2(x) = \begin{cases} 1 - |x| & |x| < 1 \\ 0, & |x| \geq 1 \end{cases}$$

$$(ii) \quad k_4(x) = \frac{1}{\sqrt{2\pi}} e^{-x^2/2}.$$

In each case the usual estimator

$$\hat{f}_m(x) = \frac{1}{Nh} \sum_{i=1}^N k((x - X_i)/h), \quad h = 2^{-m}$$

with window width h were used. The optimal window size was chosen by the method of Chiu (1991).

Wavelet estimators were based on the wavelets with scaling function given by (3.2) and (3.3). The two choices of the scaling function made were, Meyer scaling function ϕ_M and Daubechies scaling function $\phi_D = {}_2\phi$ and ${}_9\phi$, in the notation of Daubechies (1988). The estimators were then given by (2.1) or equivalently (2.2) or (2.3). Since the parameter m has discrete values, it was possible to try all reasonable values and choose the one which seemed to give the best smoothing. An alternate approach, which however we did not use, would be to start at the finest scale of interest and work backwards to a smoother version. At each stage the error is given by the wavelet coefficients. The process is continued as long as the error is small.

The results of some of our simulations are shown in *Figures 1 to 20*. We would expect the wavelet and kernel estimators both to estimate the smooth densities f_3 and f_4 well and they do. But the localization property of the wavelets should have made them superior estimators for f_1 and f_2 . This is not apparent from the graphs. In *Figures 1 – 4*, the the four densities are estimated by the kernel method using k_2 . The estimators picked up the two modes in *Figure – 4* and the bumps in *Figure – 3* quite well. However in the case of *Figure – 1* the peak at the origin was not picked up while in *Figure – 2* the two modes were not picked up either.

The results for the kernel density estimator based on k_4 are shown in *Figures 5 – 8*. The results are almost identical to *figures 1 – 4* except the estimators are somewhat smoother.

The wavelet estimators using the smooth Daubechies scaling function ${}_9\phi$ are shown in *Figures 9 – 12*. The normal densities are estimated quite well except that a slight oscillation has been introduced. In addition, in the double exponential density, the peak at 0 is picked up a little better than with the kernel based estimators. The two modes of the density in *Figure – 10* are reflected in the estimator but so are others which should not have been there.

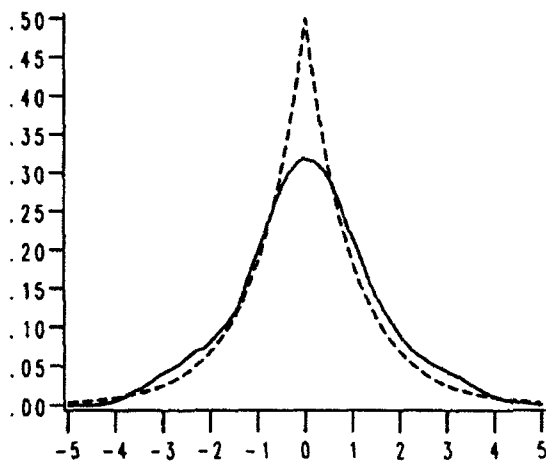
The wavelet estimator using the Daubechies scaling function ${}_2\phi$ are shown in *Figures 13 – 16*. This scaling function, though continuous, is highly irregular as can be seen from the graphs. It does pick up the peak in *Figure – 13* and *14* much better than other estimators. However it does also introduce a number of spurious oscillations which detracts from its usefulness.

In the case of the scaling function for Meyer wavelets, the results are shown in *Figures 17 – 20*. Again the estimator is better at picking up the peaks in the two densities with double exponential tails. However the estimators of the two normal densities

have much greater spurious oscillations than even in the case of ${}_2\phi$.

The oscillations could be reduced by choosing m smaller but then they fail to pick up the peaks. However by allowing the m to vary with x , one should be able to control the oscillation and pick up the peaks simultaneously. This is particularly easy to do with wavelets since the errors in going from one value of m to the next are given in terms of the wavelets themselves. This remains to be explored.

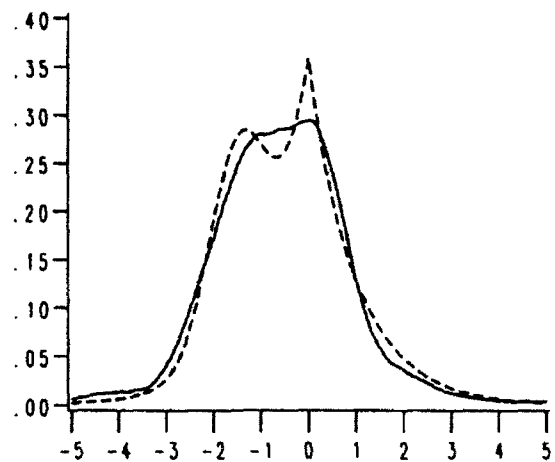
Double Exponential Density

 $n=200$, $h=0.99899$, kernel 2

True Density = Dotted Line
 Estimated Density = Solid Line

Fig - 1

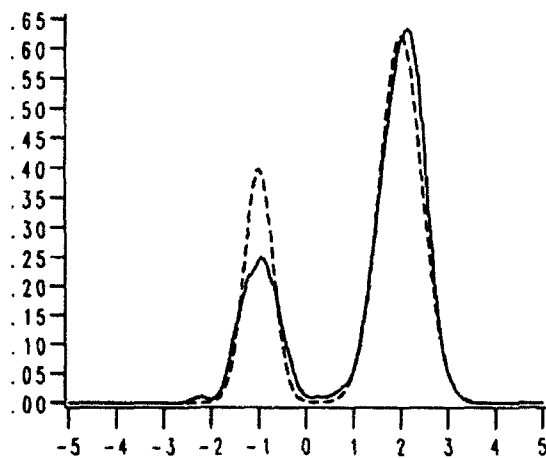
Mixture of Dexp and Normal

 $n=200$, $h=.99899$, Kernel 2

True Density = Dotted Line
 Estimated Density = Solid Line

Fig - 2

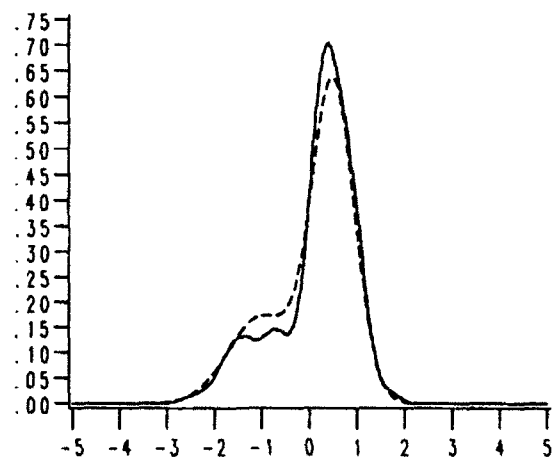
Mixture of Two Normal 2

 $n=200$, $h=.444$, Kernel 2

True Density = Dotted Line
 Estimated Density = Solid Line

Fig - 4

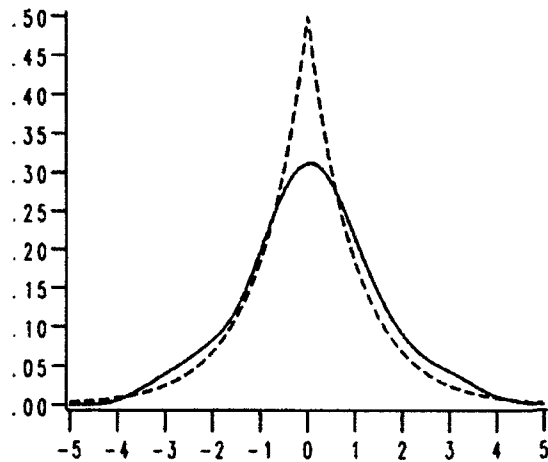
Mixture of Two Normal 1

 $n=200$, $h=.443$, Kernel 2

True Density = Dotted Line
 Estimated Density = Solid Line

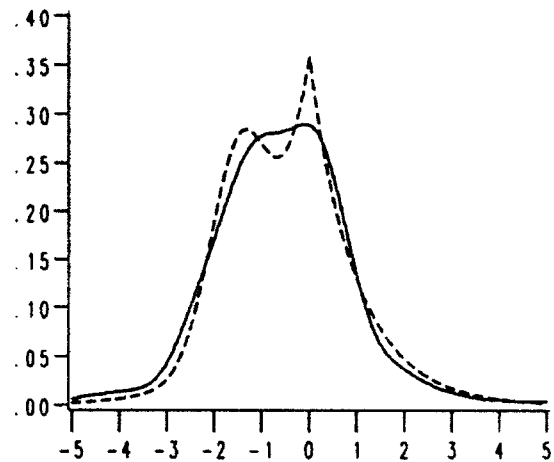
Fig - 3

Double Exponential Density

 $n=200, h=0.463, \text{kernel } 4$ 

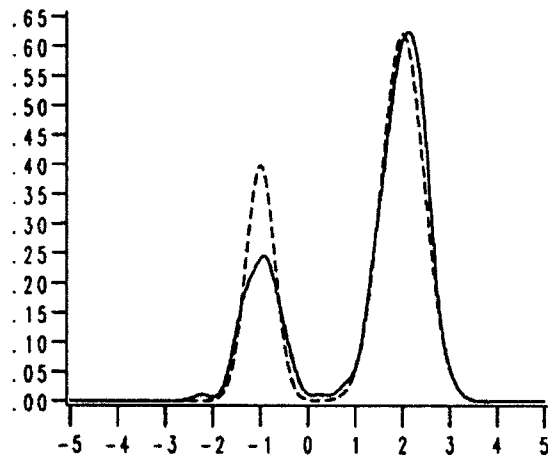
True Density = Dotted Line
 Estimated Density = Solid Line
 Fig - 5

Mixture of Dexp and Normal

 $n=200, h=.467, \text{Kernel } 4$ 

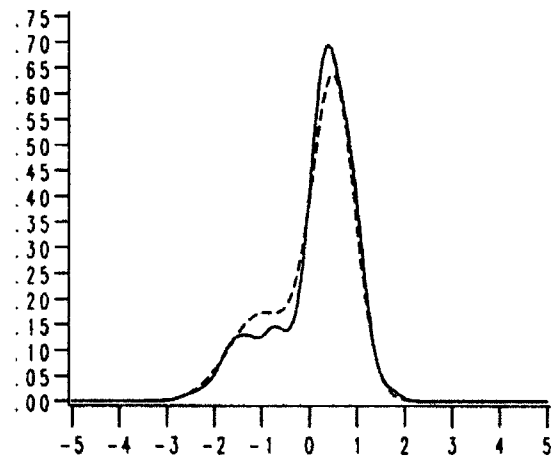
True Density = Dotted Line
 Estimated Density = Solid Line
 Fig - 6

Mixture of Two Normal 2

 $n=200, h=.191, \text{Kernel } 4$ 

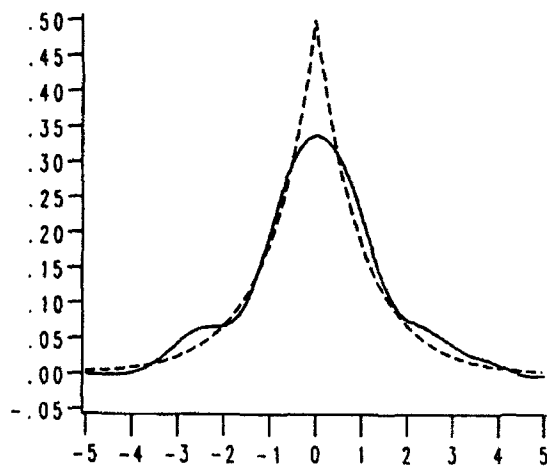
True Density = Dotted Line
 Estimated Density = Solid Line
 Fig - 8

Mixture of Two Normal 1

 $n=200, h=.195, \text{Kernel } 4$ 

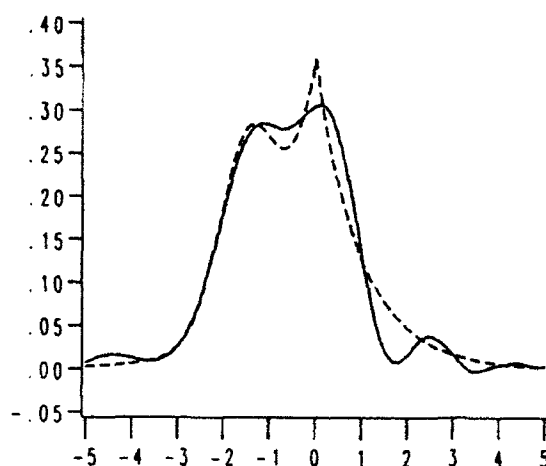
True Density = Dotted Line
 Estimated Density = Solid Line
 Fig - 7

Double Exponential Density
Daubechies(9), $n=200$, $m=0$



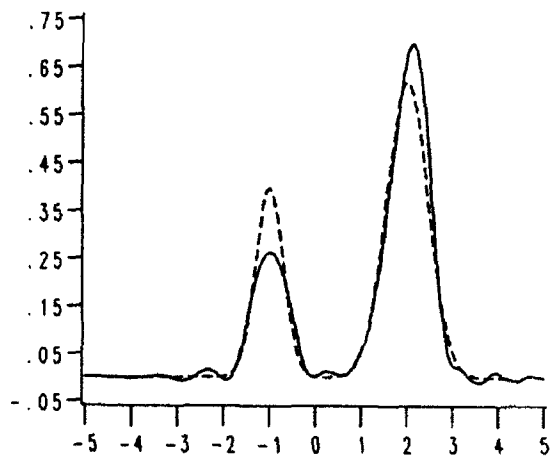
True Density = Dotted Line
Estimated Density = Solid Line
Fig - 9

Mixture of Dex and Normal
Daubechies(9), $n=200$, $m=0$



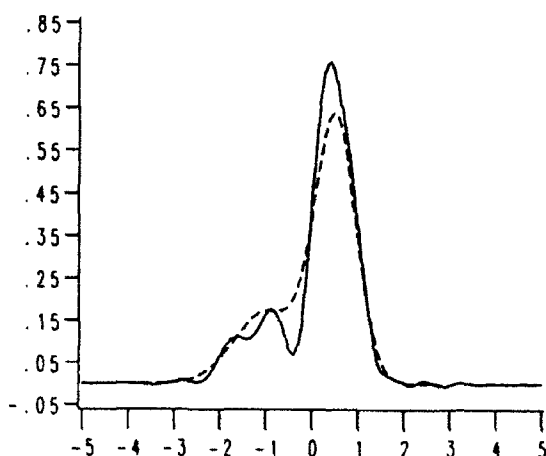
True Density = Dotted Line
Estimated Density = Solid Line
Fig - 10

Mixture of Two Normal 2
Daubechies(9), $n=200$, $m=1$



True Density = Dotted Line
Estimated Density = Solid Line
Fig - 12

Mixture of Two Normal 1
Daubechies(9), $n=200$, $m=1$



True Density = Dotted Line
Estimated Density = Solid Line
Fig - 11

Double Exponential Density

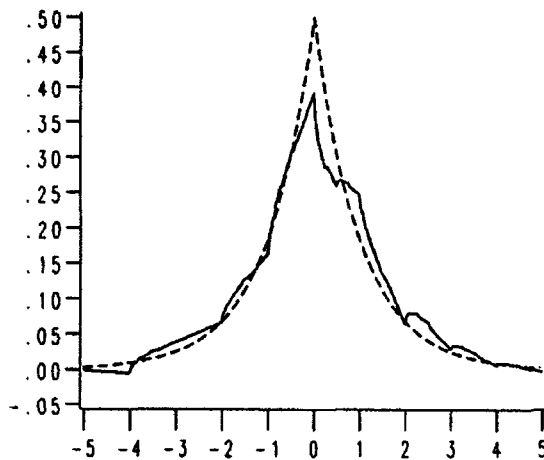
Daubechies(2), $n=200$, $m=0$ True Density = Dotted Line
Estimated Density = Solid Line

Fig - 13

Mixture of Dexp and Normal

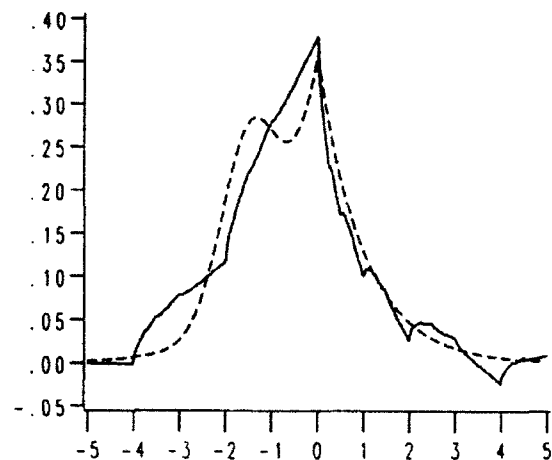
Daubechies(2), $n=200$, $m=-1$ True Density = Dotted Line
Estimated Density = Solid Line

Fig - 14

Mixture of Two Normal 2

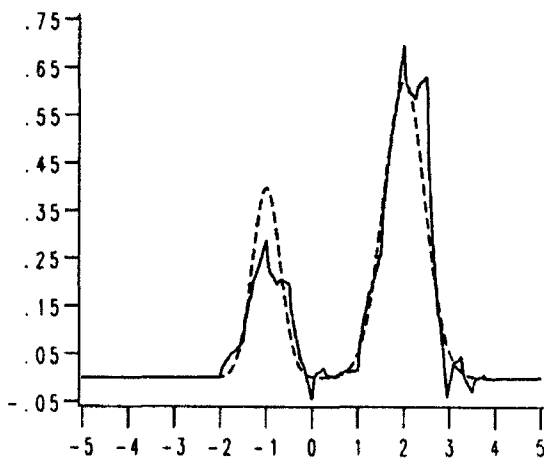
Daubechies(2), $n=200$, $m=1$ True Density = Dotted Line
Estimated Density = Solid Line

Fig - 16

Mixture of Two Normal 1

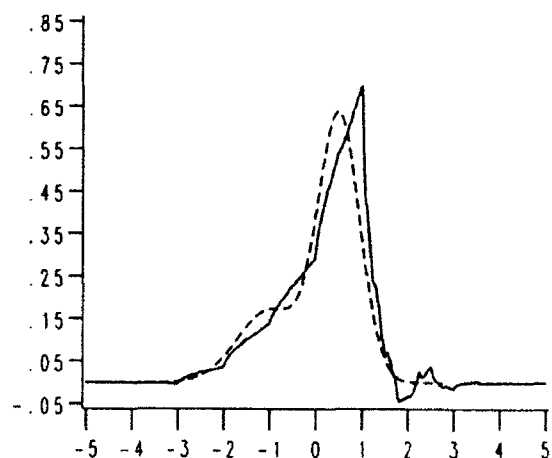
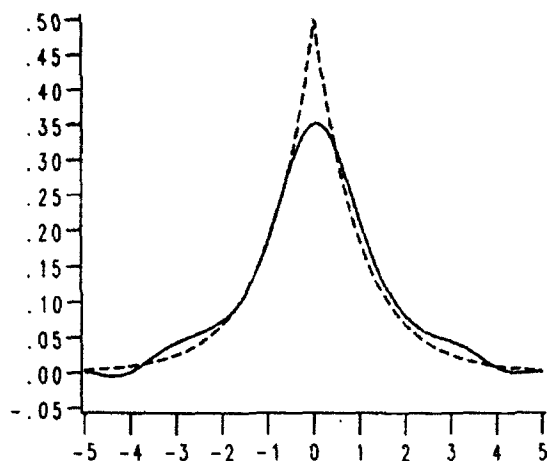
Daubechies(2), $n=200$, $m=0$ True Density = Dotted Line
Estimated Density = Solid Line

Fig - 15

Double Exponential Density

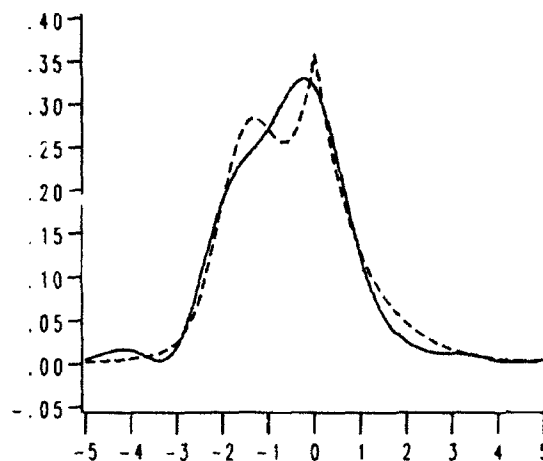
Meyer, $n=200$, $m=0$



True Density = Dotted Line
Estimated Density = Solid Line
Fig - 17

Mixture of Dexp and Normal

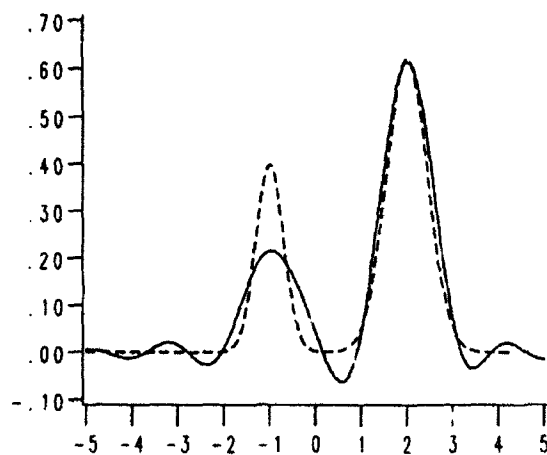
Meyer, $n=200$, $m=0$



True Density = Dotted Line
Estimated Density = Solid Line
Fig - 18

Mixture of Two Normal 2

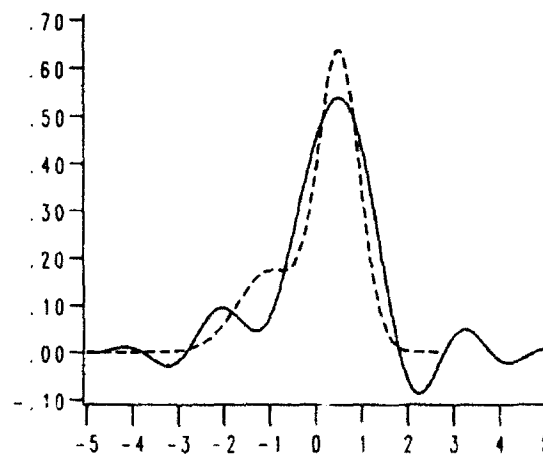
Meyer, $n=200$, $m=0$



True Density = Dotted Line
Estimated Density = Solid Line
Fig - 20

Mixture of Two Normal 1

Meyer, $n=200$, $m=0$



True Density = Dotted Line
Estimated Density = Solid Line
Fig - 19

References

- Beylkin, G., Coifman, R., and Rokhlin, V. (1989). The fast wavelet transform. Technical report, Preprint.
- Chiu, S. T. (1991). Band width selection for kernel density estimation. *Ann. Statist.*, 19:1883–1905.
- Daubechies, I. (1988). Orthogonal bases of compactly supported wavelets. *Communications on Pure and Applied Mathematics*, 41:909–996.
- Davis, K. (1977). Mean integrated squared error properties of density estimates. *Ann. Statist.*, 5:530–535.
- Földes, A. and Révész, P. (1974). A general method of density estimation. *Studia Sci. Math. Hungar.*, 9:82–92.
- Grossman, A. and Morlet, J. (1984). Decomposition of hardy functions into square integrable wavelets of constant shape. *SIAM J. Math. Anal.*, 15.
- Heil, C. and Walnut, D. F. (1989). Continuous and discrete wavelet transforms. *SIAM Review*, 31:628–666.
- Kronmal, R. and Tarter, M. (1968). The estimation of probability densities and cumulative by fourier series methods. *J. Amer. Statist. Assoc.*, 63:925–952.
- Mallat, S. (1989). Multiresolution approximations and wavelet orthogonal bases of $L^2(\mathcal{R})$. *Trans. AMS*, 315:69–87.
- Meyer, Y. (1988). The Franklin wavelets. Technical report, Preprint.
- Morlet, J., Arens, I. F., and Giard, D. (1982). Wave propagation and sampling theory, Part II. *Geophysics*, 47:203–236.
- Parzen, E. (1962). On estimation of a probability density function and mode. *Ann. Math. Statist.*, 33:1065–1076.
- Rosenblatt, M. (1956). Remark on some nonparametric estimates of a density function. *Annals Math. Statist.*, 27:832–837.
- Schwartz, S. C. (1967). Estimation of a probability density by an orthogonal series. *Ann. Math. Stat.*, 38:1261–1265.
- Strang, G. (1989). Wavelets and dilation equation. *SIAM Review*, 31:614–627.
- Wahba, G. (1975). Interpolating spline methods for density estimation I. equi-spaced knots. *Ann. Statist.*, 3:30–48.
- Walter, G. G. (1991). Approximation of the delta function by wavelets, to appear in. *J. Approx. Theory*.
- Walter, G. G. and Blum, J. R. (1979). Probability density estimation using delta sequences. *Ann. Statist.*, 7:328–340.

Investigating the Linearity of a Nonlinear Function

Lorraine Denby and James M. Landwehr

*AT&T Bell Laboratories
Murray Hill, NJ 07974*

Abstract

Consider a fitted predictive function or response surface that is nonlinear in the explanatory variables. Nevertheless, for certain values of the fitted parameters and for some region in the space of explanatory variables, the function might in fact be effectively linear with no interactions and therefore more easily interpretable. We propose and study some graphical displays to investigate this question in the context of a problem where the explanatory variables take on discrete values. We use a series of graphs showing the fitted function against each explanatory variable. The graphs are calculated conditionally on certain specified combinations of the other explanatory variables, where the combinations are based on the data from which the model was fit or the region in which predictions are desired. The methods are illustrated on a problem with seven explanatory variables and approximately 800 cases.

1. Introduction

Suppose we have a nonlinear function of several categorical variables. Such functions can arise, for example, as a response surface from fitting a statistical model to categorical data, or from the observed responses from a computer experiment run at some combination of categorical design variables. This paper presents several graphical displays that are useful for determining whether the given function is nearly linear in some region of interest. We consider the situation in which the explanatory variables are discrete, so linearity of the function implies that it has additive effects in the individual explanatory variables. Thus, in effect, we are investigating the extent to which the given function is additive in the variables on which it is defined.

This work was motivated by a direct question from a colleague requesting "approximate impact factors" for each variable in a fitted nonlinear response function. These factors can be thought of as linear approximations to the marginal effects of the levels of each variable. It would not make much sense to try to calculate such factors if the variables were not nearly additive near the combinations of explanatory variables that are possible or of interest.

Our particular function is a cost factor that was

based on three yields, namely y_p , y_o and y_r :

$$\text{cost} = \frac{y_p + y_o \cdot y_r}{y_p + y_o \cdot y_r \cdot \rho + (1 - y_p - y_o \cdot y_r) \delta}$$

where ρ and δ are given constants and each of the yields are fitted functions of seven discrete explanatory variables having from two to nine levels. The cost factor is not an observed value and cannot be fit directly to the explanatory variables, but it was developed from the fitted yields and given constants using economic principles. It is obvious from the above equation that the cost factor is a nonlinear function of the seven underlying discrete explanatory variables.

Statistical models and techniques for detecting departures from additivity have been used for some time, for example Tukey's (1949) one-degree of freedom test for nonadditivity and Mandel's (1971) modeling framework for dealing with nonadditivity in two-way tables. Numerical tests and graphical displays can help to identify types of nonadditivity within a class of models and have been studied, for example, by Marsinghe and Johnson (1981) and by Snee (1982), who uses a plot to help assess the adequacy of a certain model for interaction. DuMouchel (1988) presents an adjusted-Y plot, which is analogous to the partial residual plot, for assessing certain interactions. Cook and Weisberg (1989) suggest using three-dimensional added variable and residual plots along with dynamic graphics rotation to check for interaction.

Techniques such as these could be applied to our problem, but they were devised for the situation where the analyst is studying the response for two (or maybe three) explanatory variables and the observed response includes error. The challenge is to detect an interaction from the noisy data. In contrast, this paper considers a situation with many (here seven) explanatory variables and where the potential nonadditivity could involve any combination of them. Moreover, we are not concerned with choosing an appropriate statistical model including some random variation term, but with detecting regions where the function is effectively additive. Our plots are designed to help visualize and understand the function. We are *not* dealing with predicted or residual standard errors for the function. Section 2 describes

three plots that are useful for examining the nonadditivity. They are shown for all possible combinations of the explanatory variables. Section 3 focuses on using these plots near the region of the data that determined the function, and Section 4 gives the summary and conclusions.

2. Additivity of the Function Across All Combinations of the Variables

This paper investigates the additivity of a function of discrete explanatory variables when the underlying function is algebraically nonlinear. Additivity here is equivalent to no interactions among the explanatory variables. Therefore, our approach is to assess if and when two or more explanatory variables have an appreciable interaction by modifying ANOVA interaction plots. For the problem with two explanatory variables, the interaction plot shows c_{ij} (cost at level i for variable 1 and level j for variable 2) plotted against j with a line, or trace, connecting the points associated with a given level i of variable 1. There is one trace for each level i . If the plot displays parallel traces, then there is no interaction between variables 1 and 2. If the traces are not parallel, then there is some sort of an interaction. For examples, see Hicks (1973), pages 89-90 and 157-159.

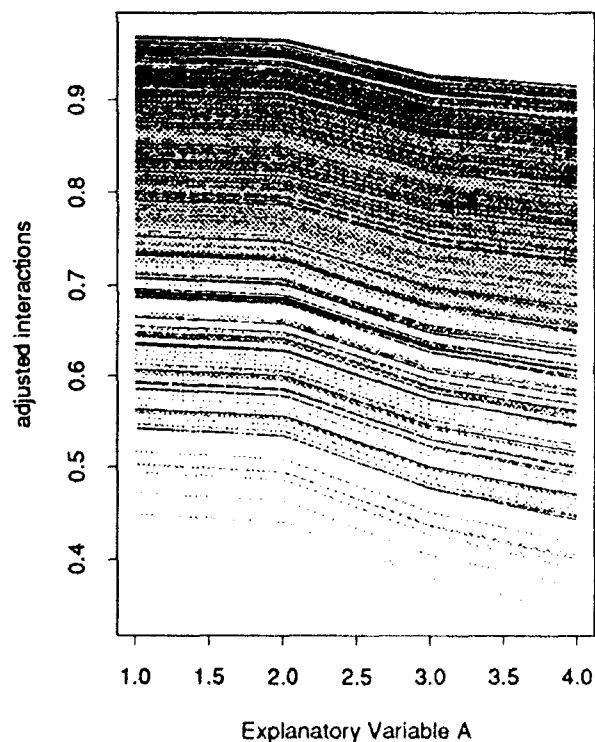


Figure 1

Throughout this paper we study the additivity of our cost function in terms of a particular explanatory variable with four levels. There are 2,592 combinations of the remaining explanatory variables, so the interaction plot shown in Figure 1 contains this many traces though they are not necessarily unique. In the notation of the previous paragraph, the subscript j corresponds to the four levels of explanatory variable A and the subscript i indexes the specific combinations of the other six variables. Figure 1 gives a strong overall impression of parallelism, but it is impossible to judge any details of departures from parallelism because there are so many traces and the plot is such a mess.

Figure 1 is related to Wegman's (1990) parallel coordinate representation for high-dimensional data, apart from being rotated by 90 degrees. Wegman represents each four-dimensional data point by a line and produces a display that looks like Figure 1; he discusses ways to use this display as a general data analysis tool for analyzing high-dimensional statistical data. In order to attack the special issues for our problem, we develop and use such displays differently than does Wegman. Jones and Rice (1992) discuss the general problem of displaying the important features of large collections of similar curves, such as in Figure 1. They propose a principal components analysis followed by display of a few particular curves that represent the original large collection.

The difficulty in assessing the degree of parallelism of the traces in Figure 1 leads us to plot $c_{ij} - \bar{c}_i + \bar{c}_{..}$ against j , which we call the adjusted interaction plot. This plot retains the main effect of explanatory variable A but eliminates the overall effect for each combination of the other explanatory variables. What we have done is move each trace i up or down so that they all have the same mean over j , namely $\bar{c}_{..}$. If there were no interactions of variable A with anything else, this plot would be a single trace. The adjusted interaction plot for our function is shown in Figure 2. This plot is not a single trace; thus, variable A does not enter our model in an exactly additive fashion over all combinations of the other explanatory variables. This plot also permits judging the size of the interactions versus the size of the main effect for this variable. Here the main effects for the different levels of variable A have a range of approximately 0.08, while the maximum spread, or interaction, is approximately 0.02. It is difficult, however, to follow an individual trace across the plot and to look for clusters of traces that are nearly parallel, which is what would be needed to learn more about the interaction structure that exists.

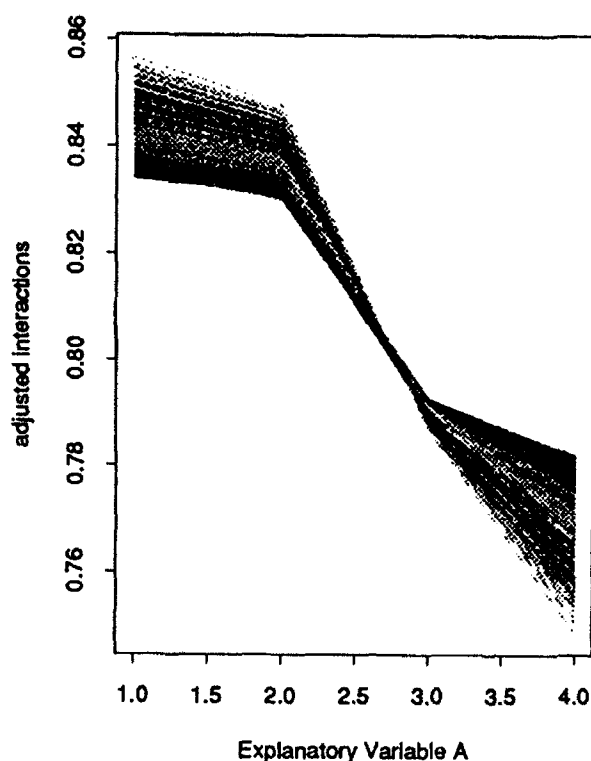


Figure 2

Individual traces can be followed more easily in a plot which we call the residual interaction plot. This plot takes the points from Figure 2 and removes the mean of each column. The residual interaction plot has $c_{ij} - \bar{c}_{i.} - \bar{c}_{.j} + \bar{c}_{..}$, the usual ANOVA interaction term, plotted against j . If there were absolutely no interactions involving explanatory variable A, this plot would be a single horizontal line at zero. Figure 3 shows the residual interaction plot for our data. A larger version of this plot suggests two clustered groups of approximately parallel traces, with one group having residual interaction greater than approximately 0.005 at level 1 of A. Using different colored line types also helps to suggest this structure.

We investigated the other explanatory variables associated with these traces using a dynamic graphics system for brushing a scatterplot matrix. Each trace is treated as a data point on four variables, where each variable represents the residual interaction at one level of variable A. Highlighting scatterplot points (each representing a trace) and identifying their levels on explanatory variables other than A revealed that the top group in Figure 3 contains exactly those combinations with level 9 of the variable with nine levels. Thus, it appears that the largest interaction with variable A involves this combination. Once this is discovered, plotting Figure 3 with all traces from level 9 of this variable in a different color

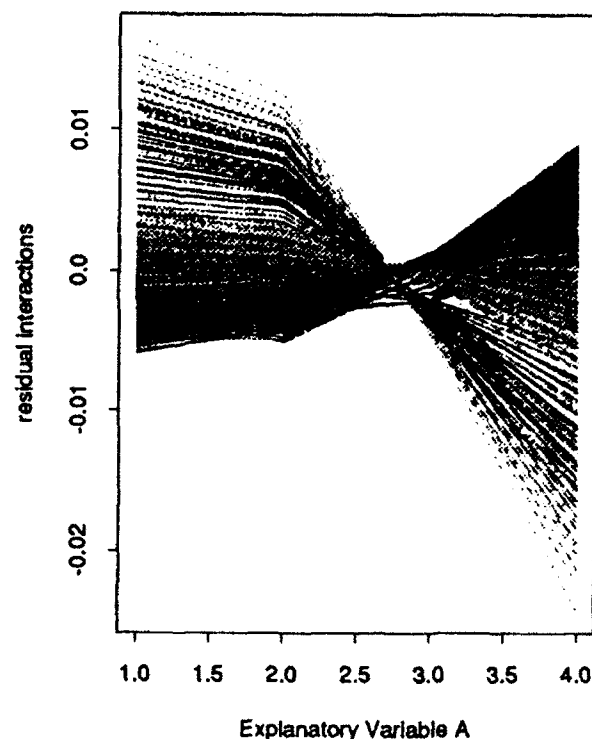


Figure 3

easily enables one to distinguish them from the others. In the next section all traces with level 9 are plotted using a dashed line type.

3. Additivity in the Data Region

Section 2 described three interaction plots calculated over all combinations of the remaining explanatory variables. If no effective interactions had been discovered, there would be no need to proceed further to understand the effects of this explanatory variable. When some effective interaction is discovered, however, an important next question is whether or not it occurs near the region containing the data from which the function was produced. Our example has 778 data points which correspond to only 143 unique combinations of the seven explanatory variables. We are primarily interested in whether or not the function is additive in the region where it would be used, which should roughly correspond to being near the region where previous data occurred. In this section we subset and adapt the three interaction plots for this purpose.

We produce the interaction, adjusted interaction, and residual interaction plots, but now showing the traces for only those combinations of the explanatory variables that occur in the data. For each such combination, we plot the trace across all possible levels of variable A, as before. Figures 4 and 5 show the

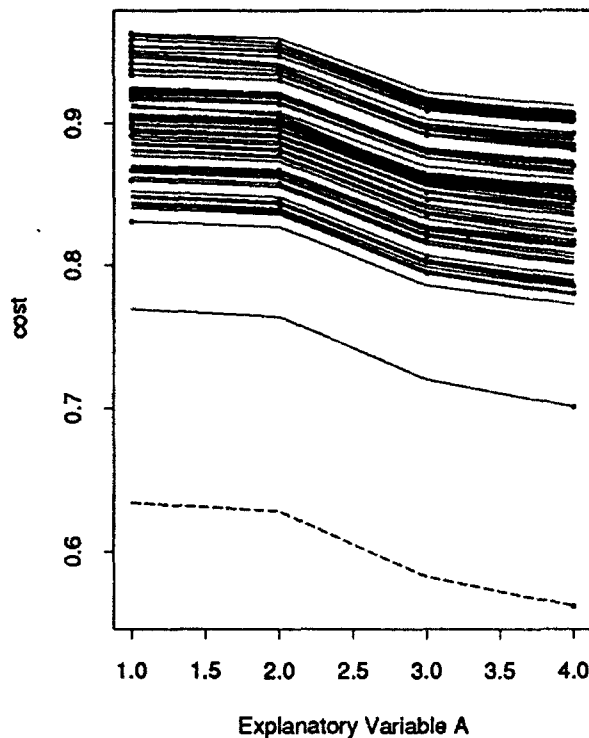


Figure 4

interaction and residual interaction plots. The points represent the actual levels of variable A. There were 143 traces used to construct these plots, though they do not all give unique lines because of the nature of the cost function.

In Figure 4 it is still difficult to assess the parallelism of the traces. We now note, however, that only two actual data values give a cost much lower than 0.8 and they have level 4 for variable A, while Figure 1 showed many such combinations. The line showing lowest cost is dashed, indicating that it is from level 9 of the identified variable. The adjusted interaction plot is not shown because of space limitations, and Figure 5 is the residual interaction plot. It shows two separate traces starting at the upper left and also suggests two distinct groups, which are apparent especially at levels 3 and 4. To interpret this structure we used dynamic graphics brushing and identification on this plot as described earlier. We discovered that the group of traces going from the lowest values for level 3 of A to middle values for level 4 correspond to a higher-order interaction involving one level of the nine-level variables, but not level 9, and three of the remaining explanatory variables. Figure 5 shows these traces as dotted lines.

Thus, from the first set of figures we discovered that the largest overall interaction with variable A involved level 9 of the identified variable, but Fig-

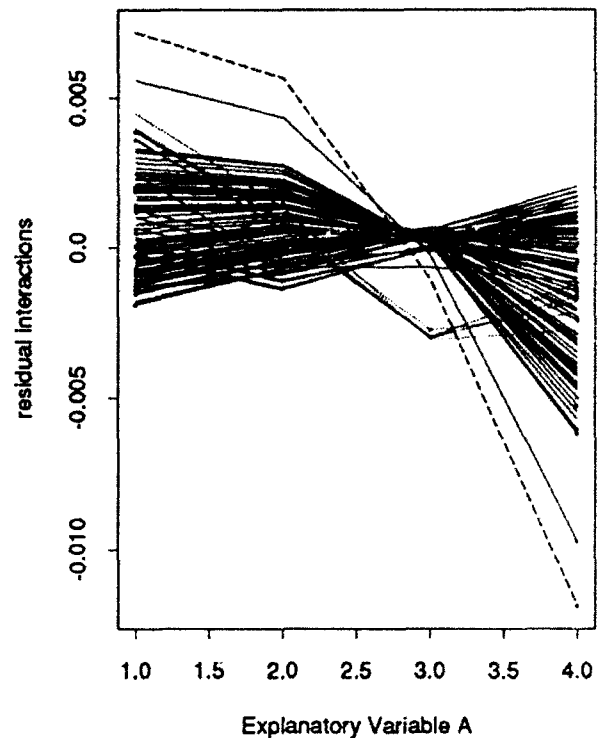


Figure 5

ures 4 and 5 showed that this interaction is not really close to most of the data since there is only one trace corresponding to level 9 exhibited in these plots. We were then able to identify an interaction of smaller magnitude that is more complicated and involves different levels of this variable, but which does occur in the region of the data.

The plots just described, however, can still show interactions in regions that could be far from the underlying data. For example, consider an actual combination with variable A at level 1; showing this trace all the way across to level 4 could suggest an interaction involving variable A at levels 3 and 4, say, that is actually relatively far from the data region at level 1. To correct for this possible problem, we modify the interaction plots to show only the segments of the traces adjacent to the actual level of this explanatory variable. This modification gives another series of three plots. The adjusted and residual segmented interaction plots cannot be centered using easy formulas as before, since now each trace does not include values at all four levels for variable A. We have an unbalanced data situation and calculate the adjustments by fitting row and column effects using linear models for unbalanced data.

Figure 6 shows the segment version of the residual interaction plot, which corresponds to Figure 3 in the first series. The actual level for variable A is

denoted by a point. Traces from combinations of the explanatory variables that have level 1 for A have a segment starting with a dot at 1 and extending to its value for level 2. Those with level 2 have a dot at its value plotted twice with a small gap above 2 and line segments extending to its values at levels 1 and 3. The other levels are handled similarly.

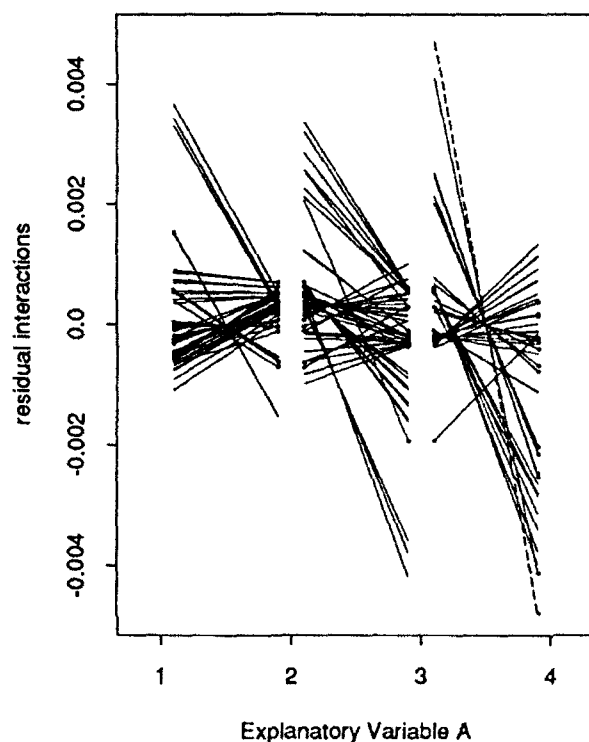


Figure 6

Note that the range for Figure 6 is only about 1/4 of the range in Figure 3, and it is also only about 1/2 of the range of the residual interaction plot from the second series, Figure 5. This observation immediately implies that there is a much smaller amount of interaction for our function in the direct neighborhood of the data than overall. Among the most noticeable segments in Figure 6, meaning those with slope farthest from horizontal, are those from level 2 extending upwards to level 1 and downwards to level 3. Investigation reveals that these segments correspond to some of the traces for the interaction identified and discussed with Figure 5. Thus, this interaction exhibits itself directly in the data region mainly through observations with level 2 on variable A. Note also in Figure 6 that the most extreme vertical slope is the one dashed line from level 4 that corresponds to the first interaction discussed with Figure 3.

4. Summary and Conclusions

We explore the nonlinearity of a function of discrete explanatory variables through three types of interaction plots, since nonlinearity in this situation reduces to nonadditivity (no interactions) for the function. The three types of plots are used to examine the function over all combinations of the explanatory variables, and also to restrict the exploration to regions near the observed data by showing appropriate subsets of the initial plots.

These plots give a practical yet powerful technique for visualizing a large and possible complicated function. For our particular cost function, only a few important interactions were identified.

References

- Cook, R.D. and Weisberg, S. Regression diagnostics with dynamic graphics, *Technometrics* 31 277-291, 1989.
- DuMouchel, W. Graphical representations of main effects and interaction effects in a polynomial regression on several predictors, *Proceedings of the 20th Symposium of the Interface of Computer Science and Statistics* 20 127-132, 1988.
- Hicks, C. R. *Fundamental Concepts in the Design of Experiments*, New York: Holt, Rinehart and Winston, 1973.
- Jones, M. C. and Rice, J. A. Displaying the important features of large collections of similar curves, *American Statistician* 46 140-145, 1992.
- Mandel, J. A new analysis of variance model for non-additive data, *Technometrics* 13 1-18, 1971.
- Marasinghe, M. G. and Johnson, D.E. Testing subhypotheses in the multiplicative interaction model, *Technometrics* 23 385-393, 1981.
- Snee, R.D. Nonadditivity in a two-way classification: Is it interaction or nonhomogeneous variance? *Journal of the American Statistical Association* 77 515-519, 1982.
- Tukey, J. W. One degree of freedom for non-additivity, *Biometrics* 5 232-242, 1949.
- Wegman, E. J. Hyperdimensional data analysis using parallel coordinates, *Journal of the American Statistical Association* 85 664-675, 1990.

Order and Influence in Regression Strategy

Julian J. Faraway

Department of Statistics, University of Michigan,
Ann Arbor, Michigan 48109, USA

Abstract

The methods (tactics) of regression data analysis such as variable selection, transformation and outlier detection are characterised as functions acting on regression models and returning regression models. The ordering of the tactics, that is the strategy, is studied. A method for the generation of acceptable models supported by the choice of regression data analysis methods is described with a view to determining if two capable statisticians may reasonably hold differing views on the same data. Optimal strategies are considered. The idea of influential points is extended from estimation to the model building process itself both quantitatively and qualitatively. The methods described are not intended for the entirely automatic analysis of data, rather to assist the statistician in examining regression data at a strategic level.

1. Introduction

Textbooks on linear regression have several chapters, each devoted to one particular aspect of building a regression model and checking its adequacy. One chapter may study variable selection and another diagnostics for the detection of outliers. If these may be viewed as tactics in the pursuit of a regression model, what of the strategy? Very little is said about how these various techniques fit together, other than that it is a skill gained by experience. Daniel & Wood(1980) is a notable exception in this respect. There are outstanding questions concerning the interaction between these tactics and the order in which they should be carried out. In this paper we will look at, not a particular specific of regression analysis, but the process as a whole.

Regression strategy is usually discussed within the context of expert systems for Statistics but production of such systems for the automatic analysis of data has been stalled primarily by the difficulty of integrating the real-world context of the data. Nevertheless, this does not preclude worthwhile study of regression strategy. This article does not describe an expert system and the tools discussed are not intended for the automatic analysis of regression data. These methods should be regarded in the same way as the usual tools of the regression analyst, such as Box-Cox transformations. The difference is

that they are designed for strategic, not tactical, application. Just as the tactical tools require the supervision of a statistician for appropriate use, these strategic tools are not meant to be blindly or automatically applied. Gale(1986) and Phelps(1987) contain several articles in which expert systems and statistical strategies are discussed.

We view the pursuit of an appropriate model by the statistician in the following way: The desired model (or models) should satisfy several requirements, unimportant variables should be excluded, outliers identified, etc. An initial model is proposed and these requirements are checked sequentially by numerical or graphical methods and if necessary the model is changed in a way suggested by the particular method. For example, variable selection methods can detect redundant variables and propose their elimination. When the current best model satisfies all the requirements the analysis ends. The choice of requirements and methods used to test these and make appropriate model changes is made by the statistician. We take particular interest in the order of application of the methods and the influence of individual data points on the final model chosen. There are two main difficulties with the tools we propose. We must automate the methods used due to the amount of repetitive analysis required. Graphically-based methods are difficult to automate because they rely on human perception and because the automated methods are context-free, the statistician must examine the results to determine the sense or lack of it. These tools are not intended to replace the standard analysis - they are an optional extra and the results should be interpreted with due care.

Regression analysis is influenced by the taste of the statistician. The choice and ordering of methods are not universally agreed upon and so it is quite possible that two experienced analysts will construct different valid models and come to different conclusions from the same data. If one was aware of this, then one would hesitate to seize upon one conclusion and discard the other, rather one might say that the data do not support any strong conclusion. However, it is quite possible that the first two statisticians may agree and a third disagree, or a fourth or a fifth. Given that the number of reasonable analyses is likely to be large, one is unlikely to have the

resources to collect all these opinions. Often, one will be the sole analyst so that it will be difficult to know if the data support many or only one conclusion.

Lubinsky & Pregibon(1987) discuss the search for "good" regression models although their methodology and motivation differ from ours. They view data analysis as the search for characteristics rather than an acceptable regression model as we do here and they describe an expert system for this search. See also Brownstone(1988) and Adams(1990).

We show a way, given a particular choice of methods, of generating all (or as many as possible) of the acceptable models arrived at by different orderings of the methods chosen. Thus the statistician may discover if there are several competing models for the data which support different conclusions, in which case suitable doubt may be expressed, or that only one model is indicated, whence the conclusion may be infused with greater confidence.

Regression analytic methods are sometimes quite inexact, perhaps depending on the statistician's interpretation of a plot, but since a large number of possible regression analyses need to be considered, these methods need to be exactly specified. This we do in section 2, so that they may be programmed. In section 3, we discuss the generation of acceptable models derived by the various orderings of the methods.

We also address the question of which of the generated models is best in section 4 and whether there is a strategy that might reliably generate this "best" model in section 5. We conclude that one should not usually pick out a best model and even if we could there is no one strategy that will surely find it.

Traditionally, influence has been defined as the effect of one point on the estimation for a given model. Using the methods discussed earlier, we can extend the idea of influence to the effect of one point on the whole process of building a regression model in section 6.

2. Characterising Regression Data Analysis

The process of building a regression model may consist of several stages such as outlier detection, variable selection and transformation which we shall call regression analytic procedures (RAPs). At each stage there is a candidate model which may be supplanted by another model according to the result of the RAP. A regression model might be specified by the original data, the functions specifying the link between the predictors and the response and weights on the observations.

Suppose we have data (X_{ij}, Y_i) with $i = 1, \dots, n$ and

$j = 1, \dots, p$, and that we consider models of the form:

$$g(Y_i) = \beta_0 + \sum_{j=1}^{p'} \beta_j f_j(X'_{ij}) + w_i \epsilon_i$$

where the primes ($'$) indicate that some initial predictor may be excluded or additional predictors like squared or interaction terms may be introduced. So p' may be greater than, less than or equal to p . The w_i are weights, which could be set to zero if we wish to exclude a point from the model. The g and f_j are transformations. Thus a regression model, in our sense, is specified by the original data, the transformations and the weights.

We wish to characterise RAP's as functions acting on regression models and returning regression models. For some procedures such as variable selection and transformation, this is relatively straightforward, but for other diagnostic based methods, it is not so easy. Thus, the RAP's provide only an approximation to a regression data analysis by a human but the information provided by this approach is additional to that given by a standard analysis, not a replacement, so nothing is lost and much may be gained. See Faraway(1992) for more discussion of this.

When constructing a regression model, we have certain requirements for what is an acceptable final choice - for example, that there be no redundant predictors, that the expected response should be linear in the predictors, that there be no outliers included in the model etc. The RAP's are a response to these requirements in that they examine a candidate model and if necessary change that model to make it acceptable with respect to that particular requirement. Thus, a minimal list of RAP's is determined by the requirements we wish our model of choice to satisfy. An acceptable final model would be one which is not changed by the application of any of the RAP in the list.

The following RAP's have been programmed. The names for the methods are given in brackets for reference. Check and remove outliers (outlier-test), Check and remove influential points (test-influence), Check for non-constant variance (hetero-test), Check for a Box-Cox transform on the response (box-cox-test), Check for transformations of the predictors (tran-predictors), Backward elimination of variables (bw-elim). Details may be found in Faraway (1992). LISP-STAT - Tierney (1990) is an ideal package for the programming of such procedures. Code is available from the author by e-mail (julian@stat.lsa.umich.edu).

This list is obviously not exhaustive but is representative of the sort of data-analytic actions that may occur in practice and is appropriate for some common requirements for acceptable regression models. Note that these

procedures allow no possibility that observations or variables excluded by the analysis at one stage may later be restored to the model and that no interaction terms will appear. This, again, is restrictive, but not importantly so. Furthermore, I am not implying that these are the best methods to use all the time only that the ideas that follow are not restricted by these particular choices.

3. Generation Of Acceptable Models

In this section we consider the generation of acceptable models by changing the order in which RAP's are applied. A natural, although arbitrary, choice of initial model is the regression of all possible predictors on the response with unit weights on the observations. Clearly, there are situations when there will be several reasonable initial choices, but for now we will use only that one initial model. Our ideas are not affected by this restriction.

Even for very small lists of RAP's, the number of sequences we must try will be very large so we must usually be satisfied with just a sample from the possibilities. Random sampling from the possible sequences may not be the best approach so we have used a systematic sampling method: Generate the $n!$ permutations of the list of RAP's and apply these cyclically to the data until each action has been applied with no change in the model. So one of these data analyses, given a list of RAP's {tran-predictors, bw-elim, outlier-test, box-cox-test}, might consist of the application of the following sequence of RAP's: {outlier-test, box-cox-test, tran-predictors, bw-elim, outlier-test, box-cox-test}, where the last change to the model occurred when box-cox-test was applied to the model for the first time. Since the final model is unchanged by any of the RAP's, we consider it acceptable.

Now, certainly some of these analyses might consist of sequences of RAP's that no statistician would ever try in practice, but the final model should be the real subject of attention and sequence of actions that generated it is of no intrinsic interest. The statistician should examine these models to ensure that they are physically sensible and discard those that are not. The difficulty is not that unreasonable models will be included amongst those considered, since the statisticians can easily screen these out, rather that important models will not be discovered due to the inflexibility of the RAP's. Thus we do not claim that this method will produce all reasonable models, but it may well find some that otherwise might have been missed. We shall use several datasets in the following discussion:

The Galapagos dataset: 29 cases being islands 5 geographic predictors (area, elevation, distance to nearest island, distance from Santa Cruz island and area of ad-

jacent island) and number of species as the response described in detail in Andrews & Herzberg (1985)

The Chicago dataset: 47 cases being zip codes in Chicago, 5 socio-economic predictors (% minority composition, fire rate, theft rate, age of housing and income and no. homeowner insurance policies is the response described in detail in Andrews & Herzberg (1985)

The Swiss dataset: 47 cases being provinces in 188 Switzerland, 5 socio-economic predictors and a standardised fertility measure is the response. Described in Mosteller & Tukey (1977)

Some data will generate several possible models, others only one. For example, using the RAP's outlier-test, test-influence, box-cox-test, tran-predictors, bw-elim, there are $5! = 120$ possible sequences. The Chicago dataset produces 7 acceptable models, the Galapagos dataset produces 5, where the transformations used, the variables included and the points excluded can all differ. In contrast the Swiss dataset produces only one acceptable model.

4. Selecting a Model

Having generated a list of acceptable models, can we choose which one is best? We have nothing new to say about model selection, rather, we are providing a wide choice of plausible models to the analyst, models that might have been discovered by hand given limitless time and patience.

Expert knowledge of the particular area may allow one to choose one model with confidence or at least eliminate some of the competitors. Some model selection methods are criterion based, like the adjusted R^2 or the Akaike information criterion. Given that the response may be transformed in different ways in the competing models the criterion may have to allow for this, so Mallow's C may be inappropriate.

We could simply pick the model that maximises the chosen criterion, but this may be precipitous. Suppose prediction is our goal then the predictions from the acceptable models may vary greatly even if the criterion does not. Given that the value of the criterion may be sensitive to small perturbations of the model, it would seem inadvisable to put too much weight on it. Also, one reason for constructing the list of acceptable models was the possibility that two capable analysts may differ in the ordering of their analysis and arrive at different final models. So it's also quite possible that using different criteria may result in different choices from the list.

Another objective in regression analysis is to assess the dependence of the response on a particular predictor. This dependence is quantified by the appropriate regression parameter. If different transformations are used

Table 1 - Analysis of Chicago data

Model	Predictors	Excluded	Adj. R^2	# Seq.	β_2	$se(\beta_2)$	P-value
1	1,2,4,5	7,24	87.0	70	-0.0172	-0.00791	3.58
2	1,2,4,5	7	83.6	10	-0.00519	0.00793	51.6
3	1,2,4, $\sqrt{4}$	24	83.3	3	-0.0306	0.00815	0.05
4	1,2,4	24	81.6	7	-0.0318	0.00854	0.06
5	1,2,4		77.9	30	-0.0197	0.00820	2.06

the acceptable models, say a log transform on the response in one model and a square root in another, it will be difficult to directly compare the relevant parameter estimates. One possibility for a consistent method of comparison is to assess the change in the response as the relevant predictor is changed (both in the original scale) at a specific point in the range of X . Alternatively, the t-statistic (or a robust version thereof) could be used as a scale-free method of comparison. Another concern in interpreting regression coefficients is collinearity, which is not specifically addressed here.

Here is an example - suppose we are interested in the dependence of the 2nd predictor, fire rate, on the response in the Chicago dataset. We perform all 120 analyses as before but restrict the RAP's from eliminating (or adding polynomial terms to) the variable fire rate. Five acceptable models are found and are described in the following table (the response is square-rooted in all these models so there is no consistent interpretation problem)

We can see that analysts might come to some very different conclusions here - no evidence of association (P-value of 51.6%), some evidence (P-values of 3.58% and 2.06%) and strong evidence (0.05% and 0.06%). It's difficult to choose between these and one shouldn't try to. It might be better that the analyst be aware that there are several possible candidate models giving different answers and that to select one of them capriciously and discard the rest would be to ignore the real uncertainty in the problem. It would be far better to report the full range of acceptable models and the estimates they make. If, however, there is only one acceptable model generated, then the analyst can be a lot more confident in the estimate.

It should be emphasised here that it would imprudent to rely on the generated models alone. We advise that the statistician perform their usual analysis without using the RAP's and paying particular attention to graphical methods and physical context. A weakness of the RAP's is they lack the human perception of graphical displays and thus may miss important features. The generated models should be regarded as additional information not as a replacement for a standard analysis.

5. A Best Strategy?

Suppose that we wished to find the model with the highest criterion value, say adjusted R^2 . A programmer of an expert system for regression data analysis might wish to know what the optimal strategy (that is ordering of RAPs) is to maximise this criterion. Consider the following example: Construct all 120 data analyses as in section 3 using the RAP's outlier-test, influence-test, box-cox-test, tran-predictors, bw-elim. Apply the same sequences to the Galapagos and the Chicago datasets and record the adjusted R^2 for each pair of analyses.

The correlation between the adjusted R^2 's is 0.38 and a plot indicates a low association between the two. Similar results have been observed with other data. This indicates that what may be a good strategy for one dataset may do poorly for another and that the best strategy (in the sense of optimising some criterion) may depend on the data itself. This has obvious implications for those who might try to automate regression data analysis in a simplistic manner - there can be no "best" strategy to fit all datasets.

6. Extending the idea of Influence

The original idea of an influential point was a point that if deleted would radically change the parameter estimates or fitted values. Cook statistics were devised to measure influence in that sense. Léger and Altman (1991) extend this idea to the influence of an observation on variable selection. We can extend this further to influence on the whole course of an analysis i.e. seeing how the selected model is changed by the elimination of one point.

Cook statistics are a popular way of measuring influence on estimation:

$$D_i = \frac{(\hat{y} - \hat{y}_{(i)})^T (\hat{y} - \hat{y}_{(i)})}{p\hat{\sigma}^2}$$

where $\hat{y}_{(i)}$ indicates the fitted value where y_i has been excluded from the estimation. We may adapt this by considering the fitted values \hat{y} after the data analysis and the $\hat{y}_{(i)}$ as the fitted values after the same data analysis

on the data with the i -th point eliminated. Note that all fitted values must be transformed back to the original scale of the response if the data analysis caused some transformation and that no economy may be made in computing these modified statistics - all n analyses must be performed as opposed to just one for the regular Cook Statistics. $\hat{\sigma}^2$ and p could be taken from the final model, although these are just scaling factors to enable the use of the F-distribution for calibration. This might well be tenuous here because $\hat{\sigma}^2$ may not be a good estimate and might not be in the right scale. In any case the relative influence will be apparent from the modified Cook's statistics even if the scale is incorrect.

To illustrate this idea, we calculate three different Cook's statistics for the Galapagos data using the data analysis sequence based on (outlier-test, box-cox-test, tran-predictors, bw-elim), where the sequence is reapplied until no further changes in the model occur. The regular Cook's statistics for the initial and final model and the modified Cook's statistics for the whole data analysis. Point 12 is the only point indicated as being influential from the Cook statistics for the final model but point 16 is the only point indicated as being influential from the other two sets of Cook Statistics. Traditional analyses might miss the influence of point 16. Point 16 is not only quantitatively influential, it also has a qualitative effect on the final model selected. The final model using all the data has two predictors and 2 outliers excluded, whereas the model for the data with point 16 excluded has an extra predictor and one (different from the final model) point excluded as an outlier.

Cook Statistics only directly measure quantitative influence but qualitative influence is also important. We can see how the datasets with one point eliminated differ from the whole data model by recording which variables, transformations and excluded points differ. For the Galapagos data only the exclusion of points 12, 16 and 25(excluded point difference only) make any difference to the form of the final model.

Traditional measures of influence only measure the effect of a point on estimation whereas this extended idea of influence can reveal more substantial perturbations caused by one point.

7. Conclusion

We have seen that the generation of all acceptable models is a useful tool in regression analysis. Practical difficulties include the complete specification of the methods of regression data analysis and the programming of RAP's in sufficient generality. The extension of the idea of influence is a useful by-product characterisation of regression data analysis. Further development of

these ideas requires a more comprehensive and versatile set of RAP's.

We recommend that the statistician do the analysis their usual manner and use the methods we have described to provide additional information. It would be unwise to rely solely on the models generated automatic procedures we have described because it is quite possible that important visible features will be missed by them and that physical context will be ignored.

We wish to emphasise that without the full incorporation of physical context into the RAP's, which is a quantum leap beyond what we have here, and without a much more comprehensive set of RAP's, the methods we have discussed here are only appropriate for careful use by statisticians and not for unguided application by the uninitiated.

References

- Adams J. (1990) *Proc. Stat. Comp. ASA*
- Andrews D. & Herzberg A (1985) "Data : a collection of problems from many fields for the student and research worker" *New York, Springer-Verlag*.
- Brownstone D. (1988) "Regression strategies" *Proceedings on the 20th Symposium on the Interface*, Ed Wegman E. et al. 74-79
- Daniel C. & Wood F. (1980) "Fitting Equations to Data, 2nd Ed." *New York, John Wiley*.
- Faraway J. (1992) "On the Cost of Data Analysis" to appear in *Journal of Computational and Graphical Statistics*
- Gale W. (Editor) (1986) "Artificial intelligence and statistics" *Addison-Wesley, Reading Mass*.
- Léger C. and Altman N. (1991) "Assessing Influence in Variable Selection Problems" *Publication #742, Département d'Informatique et de Recherche Opérationnelle, University of Montreal*.
- Lubinsky D. & Pregibon D. (1987) "Data Analysis as Search" in "Interactions in Artificial Intelligence and Statistical Methods" edited by Phelps B. Gower *Technical Press, Aldershot, Hants*
- Mosteller F. & Tukey J. (1977) "Data Analysis and Regression" *Addison-Wesley, Reading Mass*.
- Phelps B. (Editor) (1987) "Interactions in Artificial Intelligence and Statistical Methods" *Gower Technical Press, Aldershot, Hants*
- Tierney L. (1990) "Lisp-Stat: An object-oriented environment for statistical computing and dynamic graphics." *Wiley, New York*

A Gibbs Sampler Approach to the Nonlinear Mixed Model

Jack Gerson

Veterans Affairs Medical Center, San Francisco

VAMC-116R, 4150 Clement St, San Francisco, CA 94114

ABSTRACT

Monte Carlo Markov process methods based on the Gibbs sampler and the Metropolis algorithm are employed to estimate the posterior distributions of parameters in the nonlinear mixed model. A hierarchical Bayes approach is used to specify the nonlinear mixed model, enabling estimation of the posterior distributions of the variance components as well as the fixed and random effects.

The Gibbs sampler requires iterative sampling from conditional distributions, which are not always available for direct sampling in the nonlinear mixed model. To sample from such unavailable conditionals, *Gibbs-Metropolis* chains are introduced: Markov processes based on the Metropolis algorithm are nested within the Gibbs sampler iterations. This technique appears to have significant advantages over rejection sampling and ratio of uniforms. Related Markov chain Monte Carlo methods based on cycles and mixtures of Metropolis kernels are introduced; these *Metropolis-Hastings* chains retain much of the simplicity of the Gibbs sampler (moving one coordinate at a time, based on one-step conditional distributions) while requiring considerably less computing time than the Gibbs-Metropolis method.

Two applications to repeated measures data are presented.

§1. Introduction

The recent explosion of desktop computing power has made practical the everyday application of very computationally intensive statistical methods. Where in 1985 as many as 20 or more research statisticians typically developed on VAX 750's and 780's, today it is common for a statistician to have a desktop workstation with 20 to 100 times the computing power of the old VAXes. Consequently, formerly intractable problems have become tractable; the door is open to the qualitative development of new graphical and simulation methods.

In this paper, we shall explore one such application:

the use of Markov chain Monte Carlo methods to estimate posterior distributions in the nonlinear mixed model for repeated measures data.

§2. Nonlinear mixed model for repeated measure data.

By repeated measures data we mean a characteristic (the outcome) measured multiple times on each of several individuals (the observational units). Commonly in such a setup, observations within individuals are correlated, while observations between individuals are uncorrelated. When between-individual characteristics can be assumed to be normally distributed, repeated measures data is frequently handled by the linear mixed effects model (Laird and Ware, 1982), which takes the form:

$$(2.1) \quad y_i = X_i \alpha + Z_i b_i + \epsilon_i, \quad i = 1, \dots, M$$

where y_i is a (random) vector of observations for the i th individual, α is a (non-random) vector of fixed effects, b_i is a (random) vector of random effects, ϵ_i is a (random) noise vector for the i th individual, and X_i and Z_i are design matrices. Here, the b_i are assumed

to be iid $N(0, \sigma^2 D)$ and the ϵ_i independent $N(0, \sigma^2 R_i)$ where $\sigma^2 D$ and $\sigma^2 R_i$ are variance components for respectively random effects and noise.

In many instances (e.g., growth curves; pharmacokinetics), the appropriate model for longitudinal data is nonlinear in the fixed and/or random effects. Lindstrom and Bates (1990) generalized the Laird-Ware model (2.1) to a nonlinear mixed effects model for repeated measures data:

$$(2.2) \quad y_i = \eta_i(\phi_i) + \epsilon_i,$$

where $\phi_i = X_i \alpha + Z_i b_i$, the η_i are non-linear functions, and b_i and ϵ_i follow the same distributional assumptions as in (2.1).

There is no general closed form solution for the marginal posterior distribution of the random effects b_i in (2.2), since

(2.3)

$$f(b_i | y_i) = \frac{f(b_i, y_i)}{f(y_i)} = \frac{\int \int \int v d\alpha dD d\sigma^2}{\int \int \int w d\alpha dD d\sigma^2 db_i},$$

where $v = f(y_i | b_i, \alpha, \sigma^2)g(b_i | D)h(\sigma^2)j(\alpha, D)$

and $w = f(y_i | b_i, \alpha, \sigma^2)g(b_i | D)h(\sigma^2)j(\alpha, D)$,

The denominator involves the integral with respect to b_i of a term with a nonlinear exponent, which is not generally analytically integrable over b_i for nonlinear η_i .

In the remainder of this article, we employ an hierarchical Bayes extension of Lindstrom and Bates's empirical Bayes specification (2.2) and suggest alternative estimation procedures based on Markov chain Monte Carlo to solve (2.3) numerically.

§3. Markov chain Monte Carlo.

Markov chain Monte Carlo methods estimate a distribution F and functionals of F (e.g., moments) by constructing a Markov chain (or process) that converges to F , sampling from this process once suitable convergence has been achieved, and then calculating the desired estimates from the sampled values. These methods date to the Metropolis algorithm and simulated annealing (Metropolis *et al.*, 1953). Hastings (1970) generalized the Metropolis algorithm and applied it to mainstream statistical problems (Hastings, 1970). Geman and Geman (1984) presented a variant that they named the Gibbs sampler.

§3.1 The Metropolis Algorithm.

Algorithm (Metropolis): To sample from a finite-state distribution with pdf π (where π is not constant), choose any symmetric transition matrix Q , and define P by

$$(3.1) \quad p(x, y) = \alpha(x, y)q(x, y) \quad (x \neq y), \text{ where}$$

$$\alpha(x, y) = \min\left\{1, \frac{\pi(y)}{\pi(x)}\right\}$$

$$p(x, x) = 1 - \sum_{y \neq x} p(x, y).$$

The Markov chain with transition matrix P converges in distribution to π ; to sample from π , run the chain until a pre-chosen convergence criterion is met and then sample from the chain.

The Metropolis algorithm has a simple interpretation: if a candidate step has probability no less than that of the current state, always accept it; otherwise, accept the candidate step with probability equal to the ratio of the candidate probability to the current probability. Notice that it is not necessary to evaluate π ; only the ratio $\pi(y)/\pi(x)$ is needed.

§3.2 Gibbs Sampler or Metropolis Algorithm.

In its simplest form, the Gibbs sampler estimates a joint distribution $[U_1, \dots, U_m]$ by sampling round-robin from the conditional distribution of each random variable given all the others.

When exact forms cannot be computed for some of the conditional distributions, it may be difficult to obtain the exact samples needed by the Gibbs sampler. Gelfand *et al.* (1990) suggest using ratio of uniforms (Ripley, 1987). Zeger and Karim (1987) use rejection sampling (von Neumann, 1951). Unfortunately, in many applications both of these methods require unacceptably long computing times (Ripley, 1987). Instead, we apply two alternative methods:

(i) *Gibbs-Metropolis dynamics.* Construct a round-robin Gibbs sampler as follows: when conditional distributions are "available" (i.e., when a closed form can be computed), draw from that distribution. For each conditional distribution that is unavailable, construct a Metropolis chain that converges to that conditional distribution; run the Metropolis chain for j steps to obtain suitable convergence, then sample. Repeat this process at each iteration of the outer Metropolis loop. Specifics of implementation (computation of $\pi(y)/\pi(x)$; selection of auxiliary kernel Q ; choice of j) are taken up in section 4. (A similar method is discussed in Mueller (1991)).

(ii) *Metropolis-Hastings dynamics.* Hastings (1970) suggested a one-coordinate at a time approach to the Metropolis algorithm that shares much of the conceptual simplicity of the Gibbs sampler: namely, breaking a multi-dimensional transition kernel into m

one-dimensional kernels and applying the Metropolis algorithm individually to each such one-dimensional kernel. To see why this works, suppose P_1, \dots, P_m are transition matrices (or kernels), each with stationary distribution π . Then the *sequential* chain with kernel $P = P_1 P_2 \dots P_m$ and the *random* chain with kernel

$$P = \sum_{i=1}^m w_i P_i$$

(where $\sum_{i=1}^m w_i = 1$) each have stationary distribution π .

The random chain is aperiodic and irreducible

if the P_i are, and thus will converge to π whenever all the P_i do; this is not necessarily true for the sequential chain, which thus must be examined for these properties.

Note that in each round-robin cycle, Gibbs-Metropolis dynamics sample j times from each conditional distribution, while Metropolis-Hastings dynamics sample each conditional once per round-robin cycle. It is thus reasonable to consider intermediate forms (sampling each conditional five times in succession; sampling different conditionals a different number of times in succession; etc.).

§4. Applying Markov-chain Monte Carlo methods to the nonlinear mixed model.

Lindstrom and Bates specified an empirical Bayes form for the nonlinear mixed model with repeated measurements data. We adopt an hierarchical Bayes specification by adding to Lindstrom and Bates's distributional assumptions (see section 2 above) noninformative prior distributions on the variance components and the fixed effects. We also assume that the prior distributions of α and D are uncorrelated. Finally, we assume that the noise is iid—in the notation of section 2, we assume $R_i = I$.

Application of any of the methods discussed in section 3 to this model—Gibbs sampler, Gibbs-Metropolis, or Metropolis-Hastings—requires specification of the appropriate conditional distributions. For the nonlinear mixed model for repeated measures data (2.2), these conditional distributions are:

(1) the conditional distribution of the noise variance, $[\sigma^2 | \alpha, b_i, D, y_i];$

(2) the conditional distribution of the random effects variance, $[D | \alpha, b_i, \sigma^2, y_i];$

(3) the conditional distribution of the fixed effects, $[\alpha | b_i, D, \sigma^2, y_i];$

(4) the conditional distributions of the random effects, $[b_i | \alpha, D, \sigma^2, y_i];$

(Note: since we are always interested in estimating the variance component $\sigma^2 D$, rather than D alone, we henceforth let $G = \sigma^2 D$.)

Each of these conditional distributions is derived in Gerson (1992), where it is shown that (3) and (4) are not generally available. Therefore, in the applications that follow, (3) and (4) are sampled by either Gibbs-Metropolis or Metropolis-Hastings dynamics.

§5. Applications.

§5.1 Introduction.

This section applies the Markov chain Monte Carlo methods for nonlinear mixed models developed in previous sections to two data sets. The first data set contains data on tree growth, the second contains data on the uptake velocity of a chemical in the tissue of guinea pigs. We chose these data sets because they were the two examples given in Lindstrom and Bates (1990).

§5.2 Tree growth data application

Draper and Smith (1981, p.524) present data on the trunk circumference of five orange trees, each measured seven times over the course of 1582 days. Lindstrom and Bates fit this data with a nonlinear mixed effects model

$$(5.1) \quad y_{ij} = \frac{\beta_1 + b_{i1}}{1 + \beta_2 e^{\beta_3 x_{ij}}} + \epsilon_{ij},$$

where the b_i are iid $N(0, \sigma^2 D)$ and the ϵ_{ij} are iid $N(0, \sigma^2)$.

We applied a hierarchical Bayes version of (5.1), but used Metropolis-Hastings dynamics for estimation. We ran 10 chains for 1,000 iterations each, as

described in section 3. Variances were estimated adaptively, using inflation and deflation factors of respectively 1.2 and .7.

We used the density() routine from the S statistical package (Becker *et al.*, 1988) to estimate the empirical density function of each of the parameters $\beta_1, \beta_2, \beta_3, b_{i1}, \sigma^2$, and $\sigma^2 D$.

Our estimates of the posterior modes for the tree circumference data were:

$$\begin{aligned}\beta &= (196.617, 8.230, -0.00289) \\ b &= (-26.57, 27.08, -36.09, 37.46, -5.32) \\ \sigma^2 &= 61.4 \\ \sigma^2 D &= 1325.0.\end{aligned}$$

Lindstrom and Bates's estimates were:

$$\begin{aligned}\beta &= (191.184, 8.1530, -0.00290) \\ b &= (-29.51, 31.68, -37.13, 40.16, -5.20) \\ \sigma^2 &= 18.88 \\ \sigma^2 D &= 1244.9.\end{aligned}$$

§5.3 Uptake velocity data application.

Johansen (1984, p.97) published and modeled data on the uptake velocity of the chemical B-methyl-glucoside in the small intestines of guinea pigs as a function of concentration, where x_{ij} is the j th concentration level of B-methyl-glucoside in the i th guinea pig, y_{ij} is uptake velocity for the i th guinea pig at the j th concentration, ϕ_1 is maximum uptake velocity, ϕ_2 is an affinity constant and ϕ_3 is a diffusion constant.

In the data set published by Johansen, eight guinea pigs were observed at ten concentration levels. Johansen, recommends taking logarithms of both the observed velocities and the model in (5.3), which leads to the random-coefficient model

$$(6.4) \quad \ln(y_{ij}) = \ln\left(\frac{\phi_1 x_{ij}}{\phi_2 + x_{ij}} + \phi_3 x_{ij}\right) + \epsilon_{ij}.$$

Lindstrom and Bates applied a mixed model to this data, treating ϕ_1 as fixed and ϕ_2 and ϕ_3 as random. Thus, Lindstrom and Bates's model was

$$(6.5) \quad \ln(y_{ij}) = \ln\left(\frac{\beta_1 x_{ij}}{\beta_2 + b_{i2} + x_{ij}} + (\beta_3 + b_{i3})x_{ij}\right) + \epsilon_{ij},$$

where $\beta = (\beta_1, \beta_2, \beta_3)'$ is a vector of fixed effects;

$b = (b_1, \dots, b_M)'$ is a vector of random effects such that

$$b_i = (b_{i1}, b_{i2}) \text{ iid } N(0, D).$$

We applied Gibbs-Metropolis dynamics to (6.5), running 100 Gibbs chains each for 500 iterations;

Our estimates of the posterior modes for the uptake velocity data were:

$$\begin{aligned}\beta &= (0.203, 2.453, 0.00434) \\ b_1 &= (-0.164, 0.00105) \\ b_2 &= (-0.213, 0.00124) \\ b_3 &= (-0.179, 0.00101) \\ b_4 &= (0.134, -0.00078) \\ b_5 &= (-0.209, 0.00126) \\ b_6 &= (0.211, -0.00135) \\ b_7 &= (0.161, -0.00100) \\ b_8 &= (0.202, -0.00117) \\ \sigma^2 &= 0.00990 \\ \sigma^2 D &= \begin{bmatrix} 6.48 \times 10^{-2} & -3.48 \times 10^{-4} \\ -3.48 \times 10^{-4} & 2.14 \times 10^{-6} \end{bmatrix}\end{aligned}$$

Lindstrom and Bates did not publish estimates for the random effects. Their estimates of the fixed effects and variance components were:

$$\begin{aligned}\beta &= (.201, 2.393, .00445) \\ \sigma^2 &= .00908 \\ \sigma^2 D &= \begin{bmatrix} 6.07 \times 10^{-2} & -2.99 \times 10^{-4} \\ -2.99 \times 10^{-4} & 1.70 \times 10^{-6} \end{bmatrix}\end{aligned}$$

§6. Conclusion.

Future directions for this work include extending the method to (i) nonlinear mixed models with noise correlated within and/or between subjects; and (ii) nonlinear mixed models with non-Gaussian noise and/or random effects distributions.

REFERENCES

Bates, D.M. and Watts, D.G. Nonlinear Regression Analysis and Its Applications. New York: John Wiley & Sons; 1988.

Becker, P.A., Chambers, J.M. and Wilks, A.R. The New S Language. Pacific Grove: Wadsworth & Brooks/Cole; 1988.

Gelfand, A.E., Hills, S.E., Racine-Poon, A., Smith, A.F.M. Illustration of Bayesian inference in normal data models using Gibbs sampling. *Journal of the American Statistical Association*; December 1990; 85(412): 972-985.

Gelfand, Alan E. and Smith, A.F.M. Sampling-based approaches to calculating marginal densities. *Journal of the American Statistical Association*; June 1990; 85(410): 398-409.

Geman, S. and Geman, D. Stochastic Relaxation, Gibbs Distributions, and the Bayesian Restoration of Images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*; 1984; 6(6): 721-741.

Gerson, J. Estimating posterior distributions in the nonlinear mixed model using the Gibbs sampler and the Metropolis algorithm. Unpublished Ph.D. dissertation. Berkeley: Group in Biostatistics, U. of California at Berkeley; 1992.

Hastings, W.K. Monte Carlo sampling methods using Markov chains and their applications. *Biometrika*; 1970; 57(1): 97-109.

Johansen, S. Functional Relations, Random Coefficients, and Nonlinear Regression with Applications to Kinetic Data. New York: Springer-Verlag; 1984.

Laird, N.M. and Ware, J.H. Random-effects models for longitudinal data. *Biometrics*; 1982; 38: 963-974.

Lindstrom, M.J. and Bates, D.M. Nonlinear mixed effects models for repeated measures data. *Biometrics*; 1990; 46: 673-687.

Metropolis, N., Rosenbluth, A.W., Rosenbluth, M.N., Teller, A.H. and Teller, E. Equations of state calculations by fast computing machines. *Journal of Chemical Physics*; 1953; 21: 1087-1092.

Muller, P. A generic approach to posterior integration and Gibbs sampling. Technical report #91-09. Lafayette: Department of Statistics. Purdue

University; 1991.

Racine-Poon, A. Bayesian approach to nonlinear random effects models. *Biometrics*; 1985; 41: 1015-1023.

Ripley, B.D. Stochastic Simulation. New York: John Wiley & Sons; 1987.

von Neumann, J. Various techniques used in connection with random digits. *National Bureau of Standards Applied Mathematics Series*; 1951; 12: 36-38.

Zeger, S.L. and Karim, R.M. Generalized linear models with random effects: a Gibbs sampling approach. *Journal of the American Statistical Association*; 1991; 86(413): 79-86.

ESTIMATING COEFFICIENTS OF TWO-PHASE LINEAR REGRESSION MODEL WITH AUTOCORRELATED ERRORS

Tze-San Lee
Dept. of Mathematics
Western Illinois University
Macomb, IL 61455

Abstract. From the frequentist's standpoint, here is presented an iterative algorithm for estimating the unknown coefficients of two-phase linear regression model with autocorrelated errors. The case in which a change-point is unknown was considered. Applications of two-phase linear regression model with autocorrelated errors are mentioned. Finally, a numerical example using the sediment settling data was used to illustrate the proposed algorithm.

Key words: Change-Point; First-Order Markov Disturbances; Linear-Spline, Segmented, or Switching Regression Model.

1. INTRODUCTION. The two-phase (linear-spline, segmented, or switching) regression could provide a satisfactory statistical model if the underlying population which generates the data has a structural change at some point (Hudson[7]). The statistical modelling by two-phase linear regression has been applied in many areas of sciences including biology (Sprent[13]), economics (Poirier[8]), reliability engineering (Singpurwalla[11]), and medical sciences (Smith-Cook[12]). When the error terms in the model is assumed to be independently normally distributed with mean zero and constant variance, statistical inference about two-phase linear regression was studied by Hinkley[5,6] and Esterby-El-Shaarawi[4]. However, if the dependent variable in the model is monotonically increasing or decreasing for a while and then level off gradually, the assumption of independence among the error terms in the regression model is often not valid. In this case, a model assuming the correlated errors seems more plausible. The two-phase linear regression model with autocorrelated errors was studied by Salazar, et al[10] by means of Bayesian analysis. Yet no attempts from the standpoint of the frequentist were found in the literature.

It is the aim of this paper to present a frequentist's approach for estimating the

coefficients of two-phase linear regression model with autocorrelated errors. The theory is presented in Section 2. An iterative algorithm to implement the theoretical results of Section 2 is given in Section 3. Both feasible applications and a numerical example are given in Section 4.

2. THEORY. Consider the two-phase linear regression model with autocorrelated errors given by

$$y_t = \alpha_1 + \beta_1 x_t + e_t, \quad t = 1, \dots, m, \quad (2.1a)$$

$$y_t = \alpha_2 + \beta_2 x_t + e_t, \quad t = m+1, \dots, n, \quad (2.1b)$$

and e_t follows a first-order Markov process given by

$$e_t = \rho e_{t-1} + \varepsilon_t, \quad t = 2, \dots, n, \quad |\rho| < 1, \quad (2.2)$$

where $e_1 = \varepsilon_1$, the ε_t 's are measurement errors having normal distribution with $E(\varepsilon_t) = 0$ and $\text{Var}(\varepsilon_t) = \sigma^2 > 0$, $\beta_1 \neq \beta_2$, and all the parameters $\alpha_1, \alpha_2, \beta_1, \beta_2, m, \rho$ are unknown. The two straight lines of Eq. (2.1a-b) are assumed to intersect at a point with x-coordinate

$$\gamma = (\alpha_1 - \alpha_2)/(\beta_2 - \beta_1), \quad (2.3)$$

where γ is called a change-point for two-phase linear regression model. We further assume that $x_1 < x_2 < \dots < x_n$. Consequently, γ satisfies

$$x_m \leq \gamma < x_{m+1}. \quad (2.4)$$

In practical applications, γ is often unknown and has to be estimated from the data.

To estimate the parameters α 's and β 's by the method of maximum likelihood, we apply the autoregressive transformation (Cochran-Orcutt[2]) defined by

$$y_t^* = y_t - \rho y_{t-1}, \quad (2.5a)$$

$$x_t^* = x_t - \rho x_{t-1}, \quad (2.5b)$$

to Eqs. (2.1a-b). It follows that Eqs (2.1a-b) and (2.2) can be written as follows:

$$y_t^* = \alpha_1^* + \beta_1 x_t^* + \varepsilon_t, t = 2, \dots, m, \quad (2.6a)$$

$$y_{m+1}^* = \alpha_2 - \rho \alpha_1 + \beta_2 x_{m+1} - \rho \beta_1 x_m + \varepsilon_{m+1}, \quad (2.6b)$$

$$y_t^* = \alpha_2^* + \beta_2 x_t^* + \varepsilon_t, t = m+2, \dots, n, \quad (2.6c)$$

where α_1^* and α_2^* are given by

$$\alpha_1^* = (1 - \rho)\alpha_1, \quad (2.7a)$$

$$\alpha_2^* = (1 - \rho)\alpha_2. \quad (2.7b)$$

Now the estimators for the unknown α 's and β 's can be obtained by maximizing the logarithm of normal likelihood function which is equivalent to the following minimization problem:

$$\begin{aligned} \text{Minimize } \sum_{t=2,m} (y_t^* - \alpha_1^* - \beta_1 x_t^*)^2 + (y_{m+1}^* - \alpha_2 \\ + \rho \alpha_1 - \beta_2 x_{m+1} + \rho \beta_1 x_m)^2 + \sum_{t=m+2,n} (y_t^* - \alpha_2^* - \\ \beta_2 x_t^*)^2 \end{aligned} \quad (2.8)$$

subject to Eqs. (2.3-4).

To estimate the unknown ρ , an iterative procedure is set up. By letting $\rho = 0$, the constrained minimization problem of Eqs. (2.3-4,8) is reduced to the following:

$$\begin{aligned} \text{Minimize } \sum_{t=1,m} (y_t - \alpha_1 - \beta_1 x_t)^2 + \sum_{t=m+1,n} (y_t - \\ \alpha_2 - \beta_2 x_t)^2 \end{aligned} \quad (2.9)$$

subject to Eqs. (2.3-4).

Conditional on $m=M$, Eq. (2.9) can be minimized over $\alpha_1, \alpha_2, \beta_1, \beta_2$ as two separate local least squares problems. Let $\hat{\alpha}_1, \hat{\alpha}_2, \hat{\beta}_1, \hat{\beta}_2$ be the optimal solution to Eq. (2.9) and $L(\tilde{\gamma}_M)$ the corresponding residual sum of squares, where $\tilde{\gamma}_M$ is computed from substituting $\hat{\alpha}_1, \hat{\alpha}_2, \hat{\beta}_1, \hat{\beta}_2$ into Eq. (2.3). Let $\hat{\gamma}$ be the overall solution to the constrained minimization problem of Eqs. (2.3-4,8). It was shown in Hudson[7] that $\hat{\gamma}$ can be computed by noting that there exist three possible cases: (i) if $x_M < \tilde{\gamma}_M < x_{M+1}$, then $\hat{\gamma} = \tilde{\gamma}_M$, (ii) if $\tilde{\gamma}_M < x_M$, then $\hat{\gamma} = x_M$, and (iii) if $x_{M+1} < \tilde{\gamma}_M$, then

$\hat{\gamma} = x_{M+1}$. For $M = 2, \dots, n-2$, compute $L(\tilde{\gamma}_M)$. Thus, $\hat{\gamma}$ is chosen such that

$$L(\hat{\gamma}) = \min\{L(\tilde{\gamma}_M)\}_{M=2,n-2}. \quad (2.10)$$

3. ALGORITHM. An algorithm to implement the theory developed in Section 2 described as follows:

Step 1. Set $\rho = 0$. Let the overall optimal solutions to Eqs. (2.3-4,9) be denoted by $\hat{\alpha}_1, \hat{\alpha}_2, \hat{\beta}_1, \hat{\beta}_2$, and $\hat{\gamma}$.

Step 2. Compute the residuals $\hat{\varepsilon}_t$ defined as follows:

$$\hat{\varepsilon}_t = y_t - \hat{y}_t, t=1, \dots, n, \quad (3.1)$$

where \hat{y}_t is the predicted value obtained by substituting $\hat{\alpha}_1, \hat{\alpha}_2, \hat{\beta}_1, \hat{\beta}_2$ into Eqs. (2.1a-b).

Step 3. To determine whether the residuals of Eq. (3.1) random or not, plot the residuals against the predicted values and then make a judgemental decision (Draper-Smith[3]). If Eq. (3.1) are judged to be random, stop. The $\hat{\alpha}_i, \hat{\beta}_i, i = 1, 2$, and $\hat{\gamma}$ obtained in Step 1 are the desired estimates for the parameters in Eqs. (2.1a-b, 3-4). Otherwise, go to Step 4.

Step 4. Compute the maximum likelihood estimator for ρ defined by

$$\hat{\rho} = \sum_{t=2,n} \hat{\varepsilon}_t \hat{\varepsilon}_{t-1} / (\sum_{t=1,n} \hat{\varepsilon}_t^2). \quad (3.2)$$

Step 5. Set $\rho = \hat{\rho}$. By conditioning on that $m = M$, solve Eqs. (2.3-4,8). For $M = 2, n-2$, repeat a similar procedure which leads to Eq. (2.10). Go to Step 2.

Remarks.

1. If we can make a reasonable guess about the value of (x_0, y_0) , then we are able to keep the first observation in the process of autoregressive transformation. Consequently, $t = 2$ in Eqs. (2.6a,8) and (3.2) can all be replaced by $t = 1$. The

effect of keeping the first observation on the regression estimates was studied by Poirier[9].

2. The theory and algorithm proposed here are readily being extended to the case of more than one change-point occurring in the regression model. All we have to do is to apply the algorithm in a sequential way, i.e., first finding the first change-point, then a second change-point, so on and so forth.

4. APPLICATIONS. The proposed algorithm was first applied in modelling the anti-G valve data collected by the Crew Technology Division of the U.S. Air Force School of Aerospace Medicine (Burton, et al[1]). For security reasons, the anti-G valve data can not be published. Other feasible applications include the growth curve modelling, nonlinear saturable dose-response models, and calculating a practical dose threshold in radiation carcinogenesis.

To illustrate the proposed algorithm, a sediment settling data taken from Watts-Bacon[14] is used as an example. This data set was analyzed and found that the measurement errors were serially correlated. By examining the scatter plot of the data points, it was noted that there likely existed two change-points. A three-phase linear regression model with autocorrelated disturbances was tentatively proposed as follows:

$$\alpha_1 + \beta_1 x_t + e_t, t = 1, \dots, m_1, \quad (4.1a)$$

$$y_t = \{ \alpha_2 + \beta_2 x_t + e_t, t = m_1 + 1, \dots, m_2, \quad (4.1b)$$

$$\alpha_3 + e_t, t = m_2 + 1, \dots, n, \quad (4.1c)$$

and e_t is assumed to follow a first-order Markov process given by

$$e_t = \rho_1 e_{t-1} + \epsilon_t, t = 1, \dots, m_2, |\rho_1| < 1, \quad (4.2a)$$

$$e_t = \rho_2 e_{t-1} + \epsilon_t, t = m_2 + 1, \dots, n, |\rho_2| < 1, \quad (4.2b)$$

where x_t and y_t denote, respectively, time (minutes) and clear height (inches), (x_0, y_0) is taken as $(0, 0)$, the ϵ_t 's are normally distributed with mean zero and constant variance $\sigma^2 > 0$, the x -coordinates of the unknown change-points, γ_1 and γ_2 , satisfying the inequalities

$$x_{m1} \leq \gamma_1 < x_{m1+1}, \quad (4.3a)$$

$$x_{m2} \leq \gamma_2 < x_{m2+1}, \quad (4.3b)$$

$\beta_1 \neq \beta_2$, and all the parameters $\alpha_i, \beta_j, m_j, \gamma_j, \rho, \sigma^2$ are unknown for $i = 1, 2$ and $j = 1, 2, 3$.

Although the linear regression model of Eqs. (4.1a-c, 2a-b, 3a-b) has more than one change-point, we began, just as was commented in Section 3, by treating the entire data as if it were having a single change-point. After locating the first change-point, we then moved to seek the second change-point. A nonlinear procedure, called 'piecewise regression' in SYSTAT (Wilkinson[15]), was facilitated in the computation of unknown change-point. It took five and two iterations, respectively, in finding the first and second change-point in the model. Results of numerical calculations are given in Tables 1 and 2. The residual plots for both Tables 1 and 2 are omitted, but available upon request from the author. From Tables 1 and 2, it was found that $\hat{\rho}_1 = 0.882$, and $\hat{\rho}_2 = 0.632$. From the values of $\hat{\gamma}_1$ and $\hat{\gamma}_2$ of the last iteration in Tables 1 and 2, it was found that $\hat{m}_1 = 32$ and $\hat{m}_2 = 44$. After fitting separately the data set partitioned by $\hat{m}_1 = 32$ and $\hat{m}_2 = 44$, we obtain the following prediction model:

$$0.002 + 0.078x_t, t = 1, \dots, 32, \quad (4.4a)$$

$$\hat{y}_t = \{ 2.414 + 0.037x_t, t = 33, \dots, 44, \quad (4.4b)$$

$$5.478, t = 45, \dots, 48. \quad (4.4c)$$

The estimates for the unknown change-points, $\hat{\gamma}_1$ and $\hat{\gamma}_2$ of Eq. (4.3a-b) can be calculated from using Eqs. (4.4a-c) and found to be that $\hat{\gamma}_1 = 58.8$ and $\hat{\gamma}_2 = 82.8$. Incidentally, $(x_0, y_0) = (0, 0)$ and $(x_0, y_0) = (x_{32}, y_{32}) = (58, 4.42)$ were used, respectively, in the computation of Tables 1 and 2. In contrast to the hyperbola transition model used by Watts-Bacon[14], a three-phase linear regression model with autocorrelated errors of Eqs. (4.1a-c, 2a-b, 3a-b) seems more appealing because of its simple, yet informative, structure.

TABLE 1. Number of iterations used in finding the first change-point.

Iteration #	$\hat{\rho}$	$\hat{\alpha}_1$	$\hat{\beta}_1$	$\hat{\gamma}_1$
1	0.0	0.040	0.075	70.51
2	0.970	0.185	0.006	19.80
3	0.730	0.014	0.076	19.36
4	0.423	0.023	0.075	40.76
5	0.882	0.004	0.076	8.71

TABLE 2. Number of iterations in finding the second change-point.

Iteration #	$\hat{\rho}$	$\hat{\alpha}_2$	$\hat{\beta}_2$	$\hat{\gamma}_2$
1	0.0	2.082	0.042	79.36
2	0.632	-0.015	0.074	27.08

ACKNOWLEDGEMENT

This research was supported in part by an appointment of the author in the summer of 1989 as a summer faculty research fellow at the U.S. Air Force School of Aerospace Medicine, Brooks AFB, Texas. All computations were performed on the Macintosh computer through the use of SYSTAT.

REFERENCES

- [1] Burton, R.R., Shaffstall, R.M., and Jaggars, J.L. (1980), Development, test, and evaluation of an advanced anti-G valve for the F-15, *Journal of Aviation Space and Environmental Medicine*, May, 504-509.
- [2] Cochran, D. and Orcutt, G.H. (1949), Application of least squares regression to relationship containing autocorrelated error terms, *Journal of the American Statistical Association*, 44, 32-61.
- [3] Draper, N.R. and Smith, H. (1981), *Applied Regression Analysis* (Second Edition), John Wiley and Sons, Inc., New York.
- [4] Esterby, S.R. and El-Shaarawi, A.H. (1981), Inference about the point of change in a regression model, *Applied Statistics*, 30, 277-285.
- [5] Hinkley, D.V. (1969), Inference about the intersection in two-phase regression, *Biometrika*, 56, 495-504.
- [6] Hinkley, D.V. (1971), Inference in two-phase regression, *Journal of the American Statistical Association*, 66, 736-743.
- [7] Hudson, D.J. (1966), Fitting segmented curves whose join points have to be estimated, *Journal of the American Statistical Association*, 61, 1097-1129.
- [8] Poirier, D.J. (1976), *The Econometrics Of Structural Change*, North-Holland Publishing Company, Amsterdam.
- [9] Poirier, D.J. (1978), The effect of the first observation in regression models with first-order autoregressive disturbances, *Applied Statistics*, 27, 67-68.
- [10] Salazar, D., Broemeling, L., and Chi, A. (1981), Parameter changes in regression model with autocorrelated errors, *Communications in Statistics*, A10, 1751-1758.
- [11] Singpurwalla, N.D. (1974), Estimation of the join point in a heteroscedastic regression model arising in accelerated life tests, *Communications in Statistics*, 3, 853-863.
- [12] Smith, A.F.M. and Cook, D.G. (1980), Straight lines with a change-point: A Bayesian analysis of some renal transplant data, *Applied Statistics*, 29, 180-189.
- [13] Sprent, P. (1961), Some hypothesis concerning two-phase regression lines, *Biometrics*, 17, 634-645.
- [14] Watts, D.G. and Bacon, D.W. (1974), Using any hyperbola as a transition model to fit two-regime straight-line data, *Technometrics*, 16, 369-373.
- [15] Wilkinson, L. (1989), *SYSTAT: The System for Statistics for the PC*, Version 4.0, SYSTAT, Inc., Evanston, Illinois.

Multiple Comparisons for Correlated Means Bounds for Conservatism

Paul N. Somerville
University of Central Florida
Orlando, Florida 32816 USA

Summary: A bound for the conservatism of the Tukey-Kramer procedure for the correlated case is given.

1. INTRODUCTION

Suppose we have k normal populations with unknown means $\mu_i, i = 1, 2, \dots, k$ and common variance σ^2 . Let x_1, x_2, \dots, x_k be means of samples of size n from the k populations, and let s^2 be an independent estimate of σ^2 with v degrees of freedom. Let q be the $(1 - \alpha)$ quantile of the studentized range for k means and v df. Then Tukey (1953) showed that $(1 - \alpha)$ -level simultaneous confidence intervals estimates for all pairwise difference $\mu_i - \mu_j$ are given by

$$\mu_i - \mu_j \in [(x_i - x_j) \pm qs/n^{1/2}]. \quad (1)$$

For the case where x_i is the mean of a sample of size n_i , Tukey (1953) and Kramer (1956) proposed the following $(1 - \alpha)$ -level simultaneous confidence interval estimates:

$$\mu_i - \mu_j \in [(x_i - x_j) \pm (q/2^{1/2})s(n_i^{-1} + n_j^{-1})^{1/2}] \quad (2)$$

Hayter (1984) proved the procedure conservative for all values of k . Somerville (1992) gave bounds for the conservatism of the procedure for certain limiting cases.

Extending the procedure to the case where the x_i are mutually correlated, we may write the simultaneous confidence interval estimates as

$$\begin{aligned} (\mu_i - \mu_j) &\in [(x_i - x_j) \pm q/2^{1/2}(\text{var}(x_i - x_j))^{1/2}] \\ \text{or } (\mu_i - \mu_j) &\in [(x_i - x_j)/(\text{var}(x_i - x_j))^{1/2} \pm q/2^{1/2}]. \end{aligned} \quad (3)$$

For the correlated case where $k = 3$, Brown (1984) has shown that the procedure is conservative. In this paper we give a bound for the conservatism of the procedure.

2. APPROACH

We follow the geometric approach of Somerville and Van Brackle (1989). Let \underline{x} be the column vector of means x_i . Let α_{ij} be the covariance of x_i and x_j and Σ_x be the variance covariance matrix for \underline{x} . Let $\underline{\alpha}_y$ be a vector (column) with the i th element -1 , the j th element $+1$ and the remaining elements 0. Then we may write the simultaneous confidence interval estimates as

$$(\mu_i - \mu_j) \in \left[\underline{\alpha}'_y \underline{x} \pm (q/2^{1/2}) \left(\underline{\alpha}'_y \Sigma_x \underline{\alpha}_y \right)^{1/2} \right]$$

$$\text{or } (\mu_i - \mu_j) \in \left[\underline{\alpha}'_y \underline{x} / \left(\underline{\alpha}'_y \Sigma_x \underline{\alpha}_y \right)^{1/2} \pm q/2^{1/2} \right].$$

Define

$$O_k = \begin{bmatrix} 0 & 0 & \cdots & 0 & -c_1 & c_1 \\ 0 & 0 & \cdots & -2c_2 & c_2 & c_2 \\ \vdots & \vdots & & \vdots & \vdots & \vdots \\ -(k-1)c_{k-1} & c_{k-1} & \cdots & c_{k-1} & c_{k-1} & c_{k-1} \end{bmatrix}$$

where $c_i = (i(i+1))^{-1/2}, i = 1, 2, \dots, k-1$.

Define

$$H = \begin{bmatrix} O_k \\ k^{-1/2} \mathbf{1}' \end{bmatrix}.$$

We first make the orthogonal transformation (Helmert's) $\underline{u}' = H\underline{x}$ where $\underline{u}' = \begin{pmatrix} u' & u_k \end{pmatrix}$ and \underline{u} is a column vector of $k-1$ elements. It follows directly that $\underline{\alpha}'_y \underline{x} = \underline{\alpha}'_y O_k' \underline{u}$.

Without loss of generality, set $\mu_i = 0$, $i = 1, 2, \dots, k$. The region where the simultaneous confidence interval statements are correct is then the region in the u_1, u_2, \dots, u_{k-1} space bounded by the $\binom{k}{2}$ sets of parallel hyperplanes

$$\alpha_{ij}'x / \left(\alpha_{ij}' \Sigma_x \alpha_{ij} \right)^{1/2} = \pm q/2^{1/2}. \text{ We make two more trans-}$$

formations. Let $\lambda_1, \lambda_2, \dots, \lambda_{k-1}$ be the characteristic roots of Σ_u and V be the matrix of the corresponding normalized characteristic vectors. Let $\Lambda = \text{Diag}(\lambda_1, \lambda_2, \dots, \lambda_{k-1})$. First make the orthogonal transformation $r = Vu$. Second, make

the transformation $w = \Lambda^{-1/2}r$. The first transformation results in r_1, r_2, \dots, r_{k-1} being uncorrelated, and the second results in variables w_1, w_2, \dots, w_{k-1} which are $N(0, 1)$ and mutually uncorrelated.

We may write

$$\begin{aligned} \alpha_{ij}'x &= \alpha_{ij}'O_k'u \\ &= \alpha_{ij}'O_k'V'\Lambda^{1/2}w \\ &= \alpha_{ij}'Pw \text{ where } P = O_k'V'\Lambda^{1/2}. \end{aligned}$$

Also, $\alpha_{ij}'\Sigma_x\alpha_{ij} = \alpha_{ij}'P\Sigma_wP'\alpha_{ij}$. Some straightforward algebra gives p_{ab} , the (a, b) element of P (and the coefficient of w_b in the a^{th} row of Pw) as

$$\begin{aligned} p_{ab} &= (-(k-a)c_{k-a}v_{b,k-a} + c_{k-a+1}v_{b,k-a+1} + \dots + c_{k-1}v_{b,k-1})\lambda_b^{1/2}, \quad a < k \\ p_{kb} &= (c_1v_{b1} + c_2v_{b2} + \dots + c_{k-1}v_{b,k-1})\lambda_b^{1/2}, \end{aligned} \quad (4)$$

where v_{ij} is the (i, j) element of V . Suppose for some b , $\lambda_b/\lambda_a \rightarrow \infty$, $a \neq b$. Then, the element in the a^{th} row of Pw approaches $p_{ab}w_b$. Also

$$\alpha_{ij}'P\Sigma_wP'\alpha_{ij} \rightarrow (-p_{ab} + p_{jb})^2 \quad (5)$$

The hyperplanes $\alpha_{ij}'x / \left(\alpha_{ij}'\Sigma_x\alpha_{ij} \right)^{1/2} = \pm q/2^{1/2}$ approach $w_b = \pm q/2^{1/2}$.

The region of correct bounds is the region between the two hyperplanes

$$w_b = \pm q/2^{1/2}.$$

For the region to have probability content of $1 - \alpha$, must have $z_{\alpha/2} = q/2^{1/2}$ (Σ_x known). If Σ_x is known except for the factor σ^2 which is estimated by s^2 with vdf , we may have $q = z_{\alpha/2}t_{\alpha/2}(v)$.

It is obvious from geometrical considerations that it is not possible for more than 2 sets of parallel planes each distance $q/2^{1/2}$ from the origin to have greater probability content. Thus, the solution of q for given α (and v) is the most conservative possible for correlated variables. It should be noted that the conservative value of q is the same for all values of k , and is the value of q for $k = 2$.

3. EXAMPLES

Let Σ_u be a $k-1$ by $k-1$ variance covariance matrix with elements u_{ij} and characteristic roots λ_i . Border Σ_u with elements $u_{i,k-1} = u_{k-1,i}$, and $u_{kk} > 0$ to form the matrix Σ_x . It is clear from the methods for calculation of a determinant that it is always possible to choose the border elements such that Σ_x is positive definite. Now obtain $\Sigma_x = H\Sigma_uH'$.

Example 1: Let $k = 3$, $\lambda_1 = 10,000$, $\lambda_2 = 1$. Rotate the axes by 48° in the u_1, u_2 plane. Set $u_{31} = u_{13} = u_{23} = u_{32} = 0$, $u_{33} = 10,000$. We obtain

$$\Sigma_x = \begin{pmatrix} 7015.4 & -1378.3 & 4363.0 \\ -1378.3 & 9363.5 & 2014.9 \\ 4363.0 & 2014.9 & 3622.2 \end{pmatrix}$$

The correlation matrix becomes

$$\begin{pmatrix} 1. & -.17006 & .86551 \\ -.17006 & 1. & .34698 \\ .86551 & .34698 & 1. \end{pmatrix}$$

Example 2: Let $k = 4$, $\lambda_1 = 100$, $\lambda_2 = \lambda_3 = .001$. Rotate the u_1, u_2 axes by 29° and then the u_1, u_3 axes by 72° . Set $u_{14} = u_{24} = u_{34} = u_{43} = u_{42} = u_{41} = 1$, $u_{44} = 100$. We obtain

$$\Sigma_x = \begin{pmatrix} 76.807 & 36.148 & -20.366 & 7.2384 \\ 36.148 & 27.373 & 15.176 & 21.198 \\ -20.366 & 15.176 & 64.583 & 40.605 \\ 7.2384 & 21.198 & 40.605 & 31.239 \end{pmatrix}$$

The correlation matrix becomes

$$\begin{pmatrix} 1 & .78835 & -.28917 & .14777 \\ .78835 & 1 & .36093 & .72493 \\ -.28917 & .36093 & 1 & .90402 \\ .14777 & .72493 & .90402 & 1 \end{pmatrix}$$

The region of correct bounds is bounded by 12 hyperplanes, each 1 standard unit from the origin. The direction cosines of 6 of these are given below. The other six may be obtained by multiplying each direction cosine by -1.

-1.000	-.023	.0073
1.000	-.0039	.0001
1.000	-.0009	-.0012
1.000	-.0018	.0047
1.000	.0045	.0017
1.000	.0074	.0077

It should be noted that in both the above examples, it is not at all obvious that $\sum x$ approximates a "most conservative" case.

5. CONCLUSION

We have given bounds for the conservatism of the Tukey-Kramer procedure extended to the case of correlated means.

6. REFERENCES

- Brown, L. D. (1984). A note on the Tukey-Kramer procedure for pairwise comparisons of correlated means. *Design of Experiments: Ranking and Selection*, edited by T. J. Santner and A. C. Tamhane, Marcel Dekker, Inc., 1-6.
- Hayter, A. J. (1984). A proof of the conjecture that the Tukey-Kramer multiple comparisons procedure is conservative. *Ann. Statis.*, 12, 61-75.
- Kramer, C. Y. (1956). Extensions of multiple range tests to group means with unequal numbers of replications. *Biometrics*, 12, 307-310.
- Somerville, P. N. and Van Brackle (1989). On Tukey's multiple comparison procedure when the sample sizes are unequal. *Proceedings of the 21st Symposium on the Interface*, Orlando, FL.

Somerville, P. N. (1992). On the conservatism of the Tukey-Kramer multiple comparison procedure. Submitted for publication.

Tukey, J. W. (1953). The problem of multiple comparisons. Unpublished manuscript. Princeton University, NJ.

Influence Diagnostics with the Gibbs Sampler

ROBERT E. WEISS*

Biostatistics, UCLA School of Public Health, Los Angeles, CA 90024-1772

Abstract

Gibb's sampling has opened up complicated statistical models to the possibility of exact Bayesian posterior inference. With this power has come a need for simple methods for assessing the sensitivity of posteriors to assumptions. Sensitivity to an assumption is assessed by perturbing the assumption, computing a perturbed posterior, and comparing the perturbed posterior to the unperturbed posterior. Previous Bayesian work has used a Kullback divergence between these two posteriors, and most work has concentrated on the effects of case deletion.

In this paper, the L_1 norm between posteriors is used to assess the effect of the perturbation. The L_1 norm is easier to interpret than a Kullback divergence and straightforward to compute. Either divergence can be used to assess the sensitivity of the posterior to prior specification, case deletion, predictor perturbation, and case weight perturbation. A linear regression example is examined.

Key words: Bayes Theorem, Diagnostics, Sensitivity Analysis, Influential Observations.

1 Introduction

Several methods now exist to perform numerical integration for posterior calculations. The Gibbs sampler (Gelfand and Smith 1990; Gelfand, Hills, Racine-Poon and Smith 1990), is the latest of these techniques, and it has become an important method for numerical posterior evaluation. With this power has come a need for simple methods for assessing the sensitivity of posteriors to assumptions. This article explains how to compute numerical Bayesian case influence diagnostics for models fit using the Gibbs Sampler and other numerical integration techniques.

Case deletion is the most common form of sensitivity analysis. The posterior $p(\theta|Y)$ of the parameters θ given

an $n \times 1$ data vector Y is compared to the perturbed posterior $p(\theta|Y_{(i)})$ given data omitting the i^{th} case using a distance or divergence between densities. Past research on case deletion diagnostics in Bayesian analysis has used the Kullback divergences (Kullback 1959, p. 6). Much work has concentrated on making these divergences interpretable; Johnson and Geisser (1982, 1983, and 1985) and Pettit and Smith (1985) evaluate the divergences explicitly in terms of familiar leverage and residual quantities; McCulloch (1989) suggests inspecting the Kullback divergence between simple univariate densities; Carlin and Polson (1991) compare the Kullback divergence to its expectation in repeated predictive sampling. In a different approach, Weiss and Cook (1992) suggest direct plotting of $p(\theta|Y)$ and $p(\theta|Y_{(i)})$ and give a method for reducing these densities to a single dimension in generalized non-linear models.

This paper extends this past work in several directions. First, an alternative to the Kullback divergence, the L_1 norm is considered and deemed more interpretable. Second, in addition to case deletion, prior perturbations are analyzed; and sampling variability and predictor perturbations could also be included. Third, analysis of the influence measures are shown to have interesting exploratory data analytic interpretations in terms of samples from the posterior distribution of a particular scalar quantity.

In section 2, I define perturbations and discuss a general form of Bayes theorem for perturbations as well as a marginal Bayes theorem. Section 3 discusses influence statistics and computations. Section 4 gives a brief regression example. The paper closes with discussion.

2 Perturbations and Posteriors

2.1 Bayes Theorem

The posterior $p(\theta|Y)$ is related to the prior $p(\theta)$ and the likelihood $L(\theta|Y) = \prod_{j=1}^n f(y_j|\theta, x_j)$ by Bayes Theorem

$$p(\theta|Y) = \frac{p(\theta)L(\theta|Y)}{\int p(\theta)L(\theta|Y)d\theta}, \quad (1)$$

*This research was supported by grant number CA42710-07 from the USPHS/NCI.

where $f(y_j|\theta, x_j)$ is the sampling density of the j^{th} element y_j of Y given θ and covariates x_j . The covariates may be collected in an $n \times p$ matrix X with j^{th} row x_j^T . The influence of the i^{th} case is assessed by removing its likelihood contribution from the likelihood, giving $L_{(i)}(\theta) = \prod_{j \neq i} f(y_j|\theta, x_j)$, calculating a modified posterior

$$p(\theta|Y_{(i)}) = \frac{p(\theta)L_{(i)}(\theta)}{\int p(\theta)L_{(i)}(\theta)d\theta}, \quad (2)$$

and comparing $p(\theta|Y_{(i)})$ to $p(\theta|Y)$ using a distance or divergence between posteriors.

Deleting the i^{th} case results in a perturbed likelihood $L_h(\theta) = L_{(i)}(\theta) \equiv h^*(\theta, Y)L(\theta)$, where $h^*(\theta, Y) = [f(y_i|\theta, x_i)]^{-1}$ is but one example of a perturbation. Additional examples are given in the next subsection. The perturbed posterior is related to $p(\theta|Y)$ by a Bayes theorem for perturbations. Formally, we may write

$$p_h(\theta|Y) = \frac{p(\theta|Y)h^*(\theta, Y)}{E[h^*(\theta, Y)]}, \quad (3)$$

where $E[h^*(\theta, Y)] = \int p(\theta|Y)h^*(\theta, Y)d\theta$ is the posterior expectation of $h^*(\theta, Y)$.

The perturbation acts like new information used to update the posterior. As a simple example, suppose data Y and X is collected and used to update the prior $p(\theta)$ to $p(\theta|Y)$ as in (1). Later it is discovered that experimental unit i was not a member of the population under study. There is no need to start over with $p(\theta)$, $L_{(i)}(\theta)$ and equation (2), rather (3) can be used with $h^*(\theta, Y) = [f(y_i|\theta, x_i)]^{-1}$ as the likelihood generated by this new information. From (3) it can be seen that perturbations are similar to likelihoods: two perturbations $h(\theta, Y)$ and $h^*(\theta, Y)$ are equivalent if $h^*(\theta, Y) \propto h(\theta, Y)$. For the remainder of the paper, I will work with the perturbation $h \equiv h(\theta, Y) \equiv h^*(\theta, Y)(E[h^*(\theta, Y)])^{-1}$ with posterior mean 1. It will be assumed throughout that h has the same support as the posterior $p(\theta|Y)$.

2.2 Perturbations

Other perturbations besides case deletion can be considered. When the perturbation $h(\theta, Y) \equiv h(\theta)$ is not a function of the y_i 's, it is a prior perturbation. For example, $h(\theta) \propto q(\theta)\{p(\theta)\}^{-1}$ is a perturbation that can be used to investigate the influence of different priors on the posterior. Case weight perturbation might correspond to $h(\theta, Y) \propto [f(y_i|\theta, x_i)]^{-\omega}$, for $\omega \geq 0$. Response perturbation might correspond to $h(\theta, Y) \propto [f(y_i|\theta, x_i)]^{-1}f(y_i + \omega|\theta, x_i)$ with ω a fixed constant. The effect of changing $x_{i\ell}$ to $x_{i\ell} + \omega$, where

$x_{i\ell}$ is the ℓ^{th} covariate of the i^{th} case can be assessed by choosing $h(\theta, Y) = [f(y_i|\theta, x_i)]^{-1}f(y_i|\theta, x_i + \omega e_\ell)$ with e_ℓ the vector of zeros but for a one in the ℓ^{th} place. Covariate perturbation might be of interest in models with measurement error, particularly where the extra step of modeling that error has not yet been taken.

The choice of perturbation is still something of an art. Cook (1986) discusses several perturbation schemes for linear and generalized linear models. Perturbations must be interpretable and they must be chosen to reflect plausible model misspecification. Response and predictor perturbations require continuous y_i and x_i , as well as choices for ω . Case deletion perturbations are appropriate for data sets with independent observations. In models with random effects, multivariate responses or missing data, even a simple perturbation scheme such as case deletion has several non-trivial extensions.

2.3 A Marginal Bayes Theorem

In Bayes Theorem (3), the parameter vector $\theta = (\theta_1, \theta_2)$ includes all parameters and unknown quantities including parameters of interest, θ_1 , and nuisance parameters θ_2 in the problem. In problems with missing data X_{mis} , θ_2 includes the missing data; in random effects models, θ_2 may include some or all of the random effects. To find out how the perturbation h affects $p(\theta_1|Y)$, we require a marginal Bayes Theorem. First we rewrite (3) as

$$\frac{p_h(\theta_1|Y)p_h(\theta_2|\theta_1, Y)}{p(\theta_1|Y)p(\theta_2|\theta_1, Y)} = \frac{h(\theta, Y)}{E[h(\theta, Y)]}. \quad (4)$$

Multiplying both sides by $p(\theta_2|\theta_1, Y)$ and integrating with respect to θ_2 gives a marginal Bayes theorem

$$\frac{p_h(\theta_1|Y)}{p(\theta_1|Y)} = \int h(\theta, Y)p(\theta_2|\theta_1, Y)d\theta_2 = h_1. \quad (5)$$

Equation (5) generalizes the marginal Bayes theorem of Weiss and Cook (1992). The marginal perturbation $h_1 = h_1(\theta_1, Y)$ is not the obvious marginalization of $h(\theta, Y)$.

3 Influence Assessment

3.1 Influence Statistics

The divergences of Csiszár (1969) are useful for reducing the differences between the two high dimensional densities $p(\theta|Y)$, $p_h(\theta|Y)$ to scalar summaries (Weiss and Cook, 1992). For example, the axiomatic reasoning in

Bernardo (1985, 1979) suggests the influence statistic

$$K_1(h) = \int \log \left(\frac{p_h(\theta|Y)}{p(\theta|Y)} \right) p_h(\theta|Y) d\theta.$$

Johnson and Geisser (1985) also consider

$$K_2(h) = \int \log \left(\frac{p(\theta|Y)}{p_h(\theta|Y)} \right) p(\theta|Y) d\theta$$

and Pettit and Smith (1985) and Johnson and Geisser (1985) consider as well $J = K_1(h) + K_2(h)$. The diagnostics K_1 , K_2 and J are all Kullback divergences (Kullback, 1959, p.6).

A more easily interpreted statistic is the L_1 norm

$$L_1(h) = \frac{1}{2} \int |p_h(\theta|Y) - p(\theta|Y)| d\theta,$$

The statistic $L_1(h) \in [0, 1]$ is an upper bound on the change in coverage of any posterior credible interval for any marginal of any function $\beta(\theta)$ or prediction Z due to the perturbation h . It is also an upper bound on the $L_1(h)$ statistic for any marginal density of θ or any predictive density. The proofs of these statements depend on the convexity of the function $g(a) = \frac{1}{2}|a-1|$; $L_1(h) = \int g(h(\theta))p(\theta|Y)d\theta$. The bound cannot be improved; the set $S_h = \{h(\theta) > 1\}$ is one where the difference in posterior probability content under $p(\theta|Y)$ and $p_h(\theta|Y)$ achieves the bound $L_1(h)$. The perturbation $h(\theta)$ is also a scalar random variable with a posterior density $p(h(\theta)|Y)$. It has the property

$$L_1(h) = \frac{1}{2} \int |p_h(h(\theta)|Y) - p(h(\theta)|Y)| dh(\theta).$$

Like K_1 , $L_1 = 0$ means that $p(\theta|Y) = p_h(\theta|Y)$ and the perturbation has absolutely no influence. If $L_1 = 1$, it means that the two posteriors do not overlap.

The L_1 norm can be interpreted in terms of the efficiency of generating random variables from both posteriors. Suppose that two streams of random variates are to be generated, $\{\theta\}_{a=1}^A$ from $p(\theta|Y)$ and $\{\theta_h\}_{a=1}^A$ from $p_h(\theta|Y)$. One mechanism for generating the two streams is to require as many elements as possible to be identical, that is $\theta^{(a)} = \theta_h^{(a)}$ for as many a as possible, while the remaining elements of the two sequences should have different supports. This is equivalent to writing either density as, for example,

$$p(\theta|Y) = \min(p(\theta|Y), p_h(\theta|Y)) + (p(\theta|Y) - \min(p(\theta|Y), p_h(\theta|Y))),$$

where the min function is taken at each value of θ . Now generate random variables with densities proportional to the two pieces $\min(p(\theta|Y), p_h(\theta|Y))$ and $(p(\theta|Y) - \min(p(\theta|Y), p_h(\theta|Y)))$ in proportions $1 - L_1$ and L_1 . The proportion $1 - L_1$ is the proportion of random variables that can be reused in generating the sequence for the perturbed density. Thus L_1 is direct measure of similarity of two densities.

All of the statistics K_1 , K_2 , J , and L_1 suppressing the dependence on h , are influence statistics of the class

$$I_g(h) = \int g \left(\frac{p_h(\theta|Y)}{p(\theta|Y)} \right) p(\theta|Y) d\theta.$$

Weiss and Cook (1992) discuss this class in the context of generalized nonlinear models and case deletion. For K_1 , $g(a) = a \log(a)$; for K_2 , $g(a) = -\log(a)$; and for J , $g(a) = -\log(a) + a \log(a) = (a-1) \log(a)$. All of the $g(a)$'s are convex with $g(1) = 0$, which seem to be minimal requirements for influence statistics (Weiss and Cook 1992), but see Carlin and Polson (1991) for an alternative viewpoint.

3.2 Computation

Assume that $\theta^{(a)}$, $a = 1, \dots, A$ is a random sample from the posterior $p(\theta|Y)$ as might be produced by Monte Carlo or Gibbs sampling.

The influence statistics $I_g(h)$ can be estimated by

$$\hat{I}_g(h) = A^{-1} \sum_{a=1}^A g \left(\frac{h^*(\theta^{(a)})}{\hat{E}[h^*(\theta)]} \right) \quad (6)$$

where $\hat{E}[h^*(\theta)] \equiv A^{-1} \sum h^*(\theta^{(a)})$. Inspection of (6) reveals interesting exploratory data analysis interpretations of the statistics K_2 , K_1 , and L_1 ; $\exp(-K_2)$ is the posterior geometric mean of h ; similarly, $\exp(K_1)$ is the perturbed posterior geometric mean of h ; the L_1 distance is the posterior mean absolute deviation of h from its mean. Given that $E[h] = 1$, and $h \geq 0$, then the statistics L_1 , K_1 and K_2 can be considered as different measures of either variability or of skewness.

When the density $p(\theta|Y)$ is used as an importance function for Monte Carlo sampling from $p_h(\theta|Y)$, then $h(\theta^{(a)}, Y)$ are the weights corresponding to each $\theta^{(a)}$. The statistics K_1 and L_1 have interpretations for the efficiency of the sampling. The divergence K_1 is the mean information in a single sample $\theta^{(a)}$ from $p_h(\theta|Y)$ for discriminating between $p(\theta|Y)$ and $p_h(\theta|Y)$ (Kullback 1959, p. 6). Large values of K_1 indicate that it is easy to tell the two densities apart. When A , the number of draws is fixed, or when all draws $\theta^{(a)}$ are stored

for later computation, the statistic L_1 is a possible improvement on Geweke's (1989) idea of recording the 5 largest weights to diagnose a poor fit between $p(\theta|Y)$ and $p_h(\theta|Y)$. Perturbed posterior summaries from influential perturbations will be poorly estimated when $L_1(h)$ or $K_1(h)$ are large. Similarly, large values of \hat{K}_1 , for example, indicate that \hat{K}_1 is itself poorly estimated. In practice, this does not matter greatly, as influential perturbations will still be identified; exact values of influence diagnostics are rarely needed.

3.3 Influence on a Parameter Subset

The statistics $I_g(h)$ assess influence on the posterior of θ ; θ will include all parameters plus missing data and random effects. To assess influence on a parameter subset requires calculation of $h_1(\theta_1, Y)$ as in (5). This is best calculated in closed form. The obvious numerical calculation of (5) apparently requires a Gibbs sample of θ_2 for each value $\theta_1^{(K)}$. As long as θ_1 is a subset and not a transformation of θ , this will be straightforward with Gibbs sampling. In normal theory linear regression with missing x 's and in normal linear random effects models, the missing data or random effects can be removed by exact computation. Often in situations where Gibbs sampling is tractable, $h(\theta)$ can be marginalized over one parameter since the integration in (5) will be available in closed form.

Much of the time influence on parameter subsets is not necessary. By convexity of g ,

$$I_g(h) \geq I_g(h_1) \geq 0.$$

Thus the influence of any perturbation is never greater on a subset than it is on the full parameter vector. Observations uninfluential on θ are uninfluential on θ_1 .

4 Linear Regression Example

Here I use a linear regression example to illustrate the L_1 diagnostic. The model is $Y = X\beta + \epsilon$, $\epsilon \sim N_n(0, \sigma^2 I)$, $p(\beta, \sigma^2) \propto \sigma^{-2}$, with $n = 22$ cases indexed by i , and three predictors Eaves, Windows, and Yards to be used to predict Cost, in \$1000's of dollars, to rehabilitate housing in St. Paul. The Cost variable is a contractor's estimate, while the predictor variables are the averages of three surveyors' ratings of various structural elements of the outside house and immediate vicinity. The ratings for Eaves and Windows are on an integer scale from 1 (best) to 6 (worst). Yards are either in good or bad condition, and are rated either as 2

Case	Cost	E	W	Y	L_1	K_1	K_2
1	15.783	3.00	2.00	2	0.142	0.0730	0.0641
2	12.570	1.66	2.33	3	0.0775	0.0197	0.0191
3	19.600	3.33	2.33	2	0.253	0.233	0.194
4	8.206	1.66	1.66	2	0.0877	0.0263	0.0248
5	15.333	2.33	2.33	5	0.391	0.819	0.442
6	14.955	5.00	3.00	2	0.181	0.157	0.111
7	13.710	4.33	3.00	2	0.118	0.0540	0.0467
8	11.388	2.33	2.33	3	0.0823	0.0224	0.0215
9	4.802	1.33	1.66	2	0.174	0.112	0.0944
10	12.547	3.00	2.66	2	0.0725	0.0168	0.0164
11	13.677	3.00	3.33	2	0.0879	0.0269	0.0252
12	9.683	1.33	2.33	2	0.0876	0.0269	0.0252
13	16.798	2.66	3.00	4	0.100	0.0386	0.0343
14	25.615	3.00	3.33	4	0.456	0.926	0.628
15	15.734	3.00	3.00	2	0.0947	0.0308	0.0289
16	13.510	3.00	3.00	2	0.0760	0.0187	0.0182
17	13.855	3.33	3.00	2	0.0755	0.0184	0.0179
18	3.986	2.33	1.66	2	0.304	0.373	0.284
19	5.997	2.33	2.00	2	0.146	0.0758	0.0676
20	9.778	2.00	2.66	2	0.0868	0.0259	0.0244
21	18.108	1.00	1.00	2	0.787	3.88	2.34
22	10.152	2.00	3.00	2	0.109	0.0472	0.0409

Table 1: Housing data: Case number, Cost in \$1000's, predictors E = Eaves, W = Windows and Y = Yards ratings. Columns E and W give entries truncated at two digits, but calculations used 6 digits. Columns L_1 , K_1 , and K_2 are based on a Gibbs sample of length 10000. The four largest diagnostic values are in bold face.

Var	1	2	4	10	20	40	100	200
L_1	.880	.772	.638	.448	.313	.203	.102	.056
K_1	3.16	1.94	1.20	.585	.305	.140	.040	.013
K_2	28.6	13.1	5.80	1.83	.706	.251	.056	.016

Table 2: Prior perturbation diagnostics.

(good) or 5 (bad). All are averaged over three raters to get a single rating per house.

The data and case deletion diagnostics L_1 , K_1 , and K_2 are given in table 1. All calculations were performed in Lisp-Stat (Tierney, 1990) and are based on a single Gibbs sample of size 10000. Case 21 is the most influential observation with an L_1 diagnostic of .787 and a K_1 of 3.88. The diagnostics agree on the ordering of influential cases, not surprising since they all belong to Csizsar's (1967) family of divergences.

Computations based on a subsample of size 2000 usually differed from those in table 1 by only 1 or 2 in the second significant digit. Computations for cases 5 and 21 differed the most. The maximum relative error over 22 cases for L_1 was 8%; for K_1 was 28%; and for K_2 was 16%. The maximum absolute difference was always for case 21: .034 for L_1 ; .68 for K_1 ; and .24 for K_2 .

The effects of using different proper priors for β was also studied. Consider the family of priors $p(\beta)$ with

$\beta \propto N(0, cI)$. Table 2 gives values of the three influence statistics for $c = 1, 2, 4, 10, 20, 40, 100, 200$.

5 Discussion

While the discussion in section 3.2 assumed a simple random sample $\{\theta^{(k)}\}_{k=1}^K$ from the posterior $p(\theta|Y)$, the methods apply to Markov samples, Monte Carlo importance samples (Geweke 1989) and systematic samples such as produced by Gauss-Hermite quadrature product rules (Naylor and Smith, 1982). If the sample is weighted, then formula (6) and the formula for $\hat{E}[h^*(\theta)]$ need minor adjustments to account for the weights.

The diagnostics L_1 , K_1 and K_2 generally indicate similar observations as influential. All are based on posterior expectations of functions of the perturbations $h(\theta)$. Thus the choice, if choice need be made, may be driven by other considerations. The L_1 diagnostic, it has been argued, is more interpretable. If more detailed analysis is required, then the plots of Weiss and Cook (1992) can be inspected.

References

- [1] Bernardo, J. M. (1979). Expected information as expected utility. *The Annals of Statistics* 7, 686-690.
- [2] Bernardo, J. M. (1985). In Discussion of Pettit and Smith, in *Bayesian Statistics 2*, eds. J. M. Bernardo, M. H. DeGroot, D. V. Lindley, and A. F. M. Smith, Amsterdam: North Holland, 492-493.
- [3] Carlin, B. P. & Polson, N. G. (1991). An expected utility approach to influence diagnostics. *J. Am. Statist. Assoc.*, 86, 1013-1021.
- [4] Cook, R. D. (1986). Assessment of Local Influence (with discussion). *J. R. Statist. Soc. B* 48, 133-69.
- [5] Csiszár, I. (1967). Information-type measures of difference of probability distributions and indirect observations. *Studia Scientiarum Mathematicarum Hungarica* 2, 299-318.
- [6] Gelfand, A. E. & Smith, A. F. M. (1990). Sampling-based approaches to calculating marginal densities. *J. Am. Statist. Assoc.* 85, 398-409.
- [7] Gelfand, A. E., Hills, S. E., Racine-Poon, A., & Smith, A. F. M. (1990). Illustration of Bayesian inference in normal data models using Gibbs sampling. *J. Am. Statist. Assoc.* 85, 972-985.
- [8] Geweke, J. (1989). Bayesian inference in econometric models using Monte Carlo integration. *Econometrica* 57, 1317-1339.
- [9] Johnson, W. & Geisser, S. (1982). Assessing the Predictive Influence of Observations. In *Statistics and Probability: Essays In Honor of C. R. Rao*, Eds. G. Kallianpur, P. R. Krishnaiah, J. K. Ghosh, pp. 343-358. Amsterdam: North-Holland.
- [10] Johnson, W. & Geisser, S. (1983). A Predictive View of the Detection and Characterization of Influential Observations in Regression Analysis. *J. Am. Statist. Assoc.* 78, 137-44.
- [11] Johnson, W. & Geisser, S. (1985). Estimative Influence Measures for the Multivariate General Linear Model. *J. Statist. Plan. Inf.* 11, 33-56.
- [12] Kullback, S. (1959). *Information Theory and Statistics*, Gloucester, MA: Peter Smith.
- [13] McCulloch, R. E. (1989). Local model influence. *J. Am. Statist. Assoc.* 84, 473-478.
- [14] Naylor, J. C. & Smith, A. F. M. (1982), "Applications of a Method for the Efficient Computation of Posterior Distributions," *Journal of Applied Statistics*, 31, 214-225.
- [15] Pettit, L. I. & Smith, A. F. M. (1985). Outliers and Influential Observations in Linear Models. In *Bayesian Statistics 2*, Eds. J. M. Bernardo, M. DeGroot, D. Lindley, and A. F. M. Smith, pp. 473-94. Amsterdam: North Holland.
- [16] Tierney, L. (1990). *LISP-STAT*. New York: Wiley.
- [17] Weiss, R. E. & Cook, R. D. (1992). A graphical case statistic for assessing posterior influence. *Biometrika*, 79, 51-55.

Reliability Model Generator

Andrew J. Booker

Boeing Computer Services
P.O. Box 24346, MS 7L-21
Seattle, Washington 98124-0346

Abstract

We describe experiences with a prototype tool for support of reliability modeling called Reliability Model Generator (RMG). RMG is a graphical interface between designers of systems and the existing reliability modeling tools ASSIST and ASSURE. RMG represents a system by a block diagram of the system's components and interconnections. RMG uses the block diagram and information about reliability behavior and failure effects of components to create an aggregate reliability model for the system and generate an input file for the reliability analysis tool ASSIST.

Introduction

The Reliability Model Generator (RMG) is a tool to support reliability modeling. It is a graphical interface between designers of fault tolerant systems and the existing reliability modeling tools ASSIST and ASSURE (see [7] and [8]) used to assess the reliability of these systems.

RMG represents a system by a block diagram of the system's components and interconnections. RMG uses the block diagram, reliability models of components, and information about failure effects of components on each other to generate a system reliability model and create an input file for ASSIST (see [7]). RMG has three essential parts.

1. RMG has a graphical user interface for creating components, entering reliability models and connecting components to form systems and subsystems as "building blocks". Building blocks are saved in knowledge bases accessed through windows which display component pictures and connections between components. They may be copied or copied and modified and used to form new candidate architectures.
2. A reliability model aggregation system creates and simplifies a system reliability model from component models. The aggregation algorithm is described in its original form in the RMG specification document [5], except for recent changes.
3. A model reducer/encoder generates an input file for ASSIST. RMG generates the model after prompting the user for information such as initial states of components, failure rates and system failure conditions.

What follows is a description of RMG and some example components and systems. The last two examples are a fault tolerant processor and a network from the IAPSA [4] architecture used as a test case for RMG.

A Simple Example

Figure 1 depicts a simple example of a system with components, ports and connections (see also Figure 3). There are two component types, A and B, where A1 and A2 are 2 "instances" of component type A. Output from A1 is input to B. Components in the system have either failed or not failed. Component output is characterized as "good" or "bad". Component type A transmits "good" information if it has not failed. Component type B transmits "good" information if it has not failed and if it receives "good" information. The system fails if neither B nor A2 have "good" output.

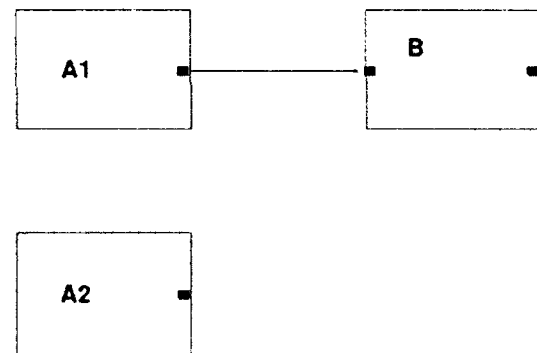


Figure 1 A Simple System

Figure 2 depicts the Markov model for the simple example system. The Markov model describes states of the system, represented by circles or "nodes", in terms of states of components. In this example component states are NOF (not-failed) and FAIL. The system changes state when a component makes a "transition", represented by the arrows. In this example the component transitions have an initial state (NOF), a final state (FAIL) and a rate (LA, LB, etc).

Note that the Markov model is relatively complicated even though the system is very simple. Evaluation of system reliability requires determination of the failure states of the system and estimation of the probability the system is in a failure state before a given time. There are tools available for evaluation of system reliability via markov models. Some of them are SURE[2], HARP[6], and CAREIII[1]. These tools

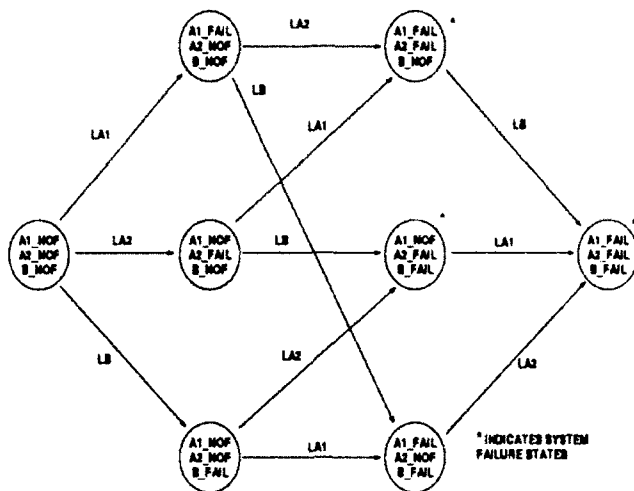


Figure 2 Markov Model

require some or part of explicit modeling and identification of failure states.

Components in RMG

This section gives the content and form of information RMG needs to describe a component. A component has a graphical representation and a model of the behavior of the component called a Local Reliability Model (LRM).

Representation of Components Figure 3 shows a graphical representation of a typical component in RMG. It is a rectangle with a component name, ports, and port labels. The user creates a component by selecting "create component" from a menu. RMG then prompts for the component name, and each port and port label. Components can have 3 types of ports: input, output, and input/output. Input/output ports are a short cut to drawing both an input and output port. The component also has states, called "modes" in RMG. They are not depicted graphically.

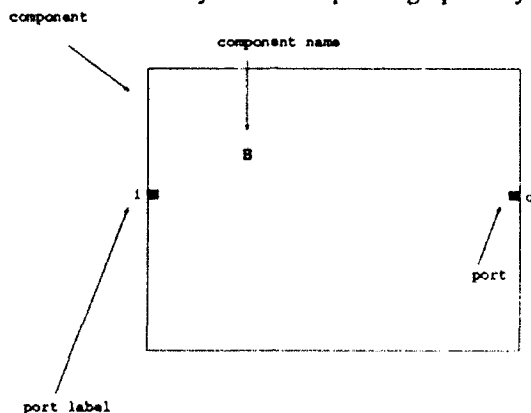


Figure 3 Graphical Representation of a Component

Input and Output Characteristics In a reliability analysis of a component, the actual values of inputs or

outputs may not be known or may not be of interest. In RMG one only specifies "characteristics" of the inputs or outputs. They are good (g), bad (b), or nil (n). These characteristics have no special meaning to RMG although the intent is for them to be used in the following way. A component's output has characteristic "g" if, for example, the component has not failed and it has not received faulty information from other components. The characteristic "b" represents corrupted output (or input) that, for example, results from a component failure. The characteristic "n" represents output which is not produced on time or output from a component which has been turned off (no output).

Conditions RMG descriptions of output characteristics, transitions and system failure are with "conditions". There are two basic types of conditions. A "port condition" represented here in the form

(port label characteristic)

and a component mode condition in the form

(component name mode).

General conditions may be conjunctions (logical "and") or disjunctions (logical "or") of the above types along with a few "predicates" used for short cuts: NUMBER (list of conditions), understood as "the number of" conditions that are true, ALL (list of conditions), or SOME (list of conditions)

Local Reliability Models A local reliability model (LRM) describes the component's output in terms of inputs and modes and its changes or "transitions" from mode to mode. A local model of a component has:

1. A list of input ports, output ports, and modes.
2. For each output port conditions under which the port has the characteristics "g", "b", or "n". This is a list of "output characteristic definitions" (OCD).
3. Conditions under which mode transitions occur.

OCD A description in RMG of each output port's characteristics in terms of conditions such as the modes of the component and input characteristics is called an "output characteristic definition" (OCD). RMG represents an OCD as an expression tree. For example, the trees representing the OCDs for components A and B are shown in Figure 4. Note that input port conditions and component modes correspond to leaves of a tree and output port conditions correspond to roots.

Transitions An LRM also has a list of component modes and transitions (from mode to mode) in the form:

IF (<cond>) TRANTO (<mode>) BY <rate>

where <cond> is a general condition, <mode> is a mode condition and <rate> may be a number or a symbolic name for a rate.

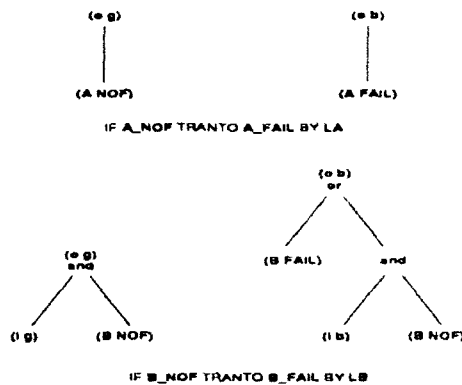


Figure 4 LRM Transitions and Tree Representation of OCD

Other Modeling Devices Some other features and capabilities were added to RMG to accommodate modeling the IAPSA architecture.. See [3] for details. **Hierarchical modeling** in the form of parent-subcomponent relationships allows one to create a high level model of the behavior of a component or subsystem and add more detail to the model as information becomes available. **Transition effects** provide a way to have transitions in any component cause a change in mode in any other component. **Component groups** allow one to model relationships among components whose physical location in the RMC representation makes it impossible or difficult to put the components inside another component or for components which may be a member of two "subsystems". Finally, very complicated transitions may be created outside of RMG then loaded into RMG and encoded into ASSIST later.

Aggregation and Simplification

There are two steps in the process of building an overall model from component LRMs and system connections. They are aggregation and simplification.

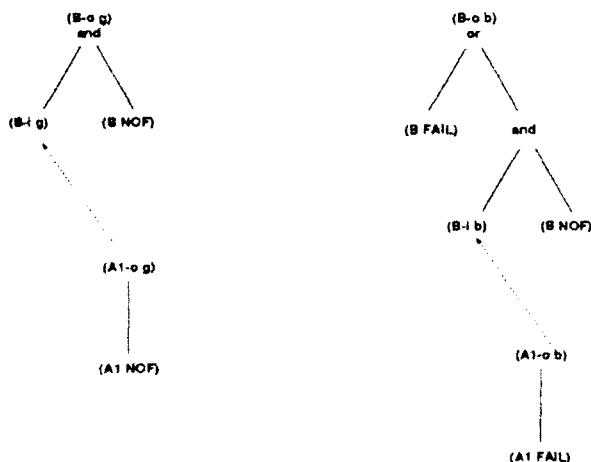


Figure 5 The Aggregation Process

The aggregation process creates system OCDs. The process gets conditions for a port characteristic from the port OCD. For each input port mentioned in the OCD, RMG looks at output ports connected to the input port in the graphical representation and substitutes from their OCDs into the conditions. Aggregation terminates when all unsubstituted conditions are on unconnected inputs, component modes or part of a cycle (see below). A condition being aggregated may also be part of a condition for transition.

Figure 5 illustrates the aggregation process in terms of OCD trees for the simple example system. Since output from A1 is connected to input to B in the RMG picture of the system, the aggregation process substituted from the component A1's OCDs for (A1-o g) and (A1-o b) for the conditions (B-i g) and (B-i b) in the OCD for B. The resulting trees are system OCDs. The aggregated model is shown in Figure 6.

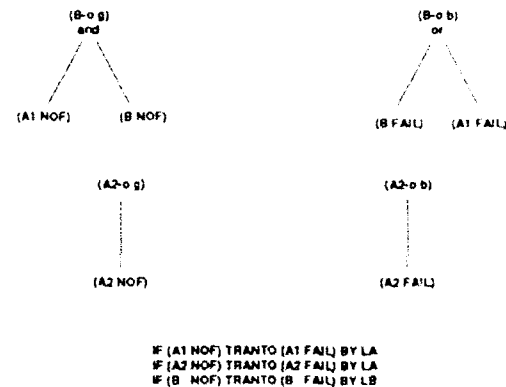


Figure 6 An Aggregated Model

Once a tree for an aggregated model is constructed, the simplification process puts the resulting OCDs for the aggregated model into disjunctive normal form (DNF). This form enables reduction of the expressions in the system OCD and removal of redundant conditions.

Examples

4 Node Network Figure 7 is a network with one root node (R), 3 other nodes (N), and 6 links (L). The lines with arrows at each end indicate that connections are made to input-output ports. The local reliability information can be described succinctly. A node or link can transmit good information if it can get it from one of its inputs. The root node must get "g" from the input labelled "i". Aggregation of this example results in a cycle. As the process constructs the expression tree, shown in Figure 8, generated when RMG traces the condition "(o g)" on node N3 a cycle occurs because the condition "(i1 g)" is repeated along a path below the first occurrence of the condition in the tree. Continued

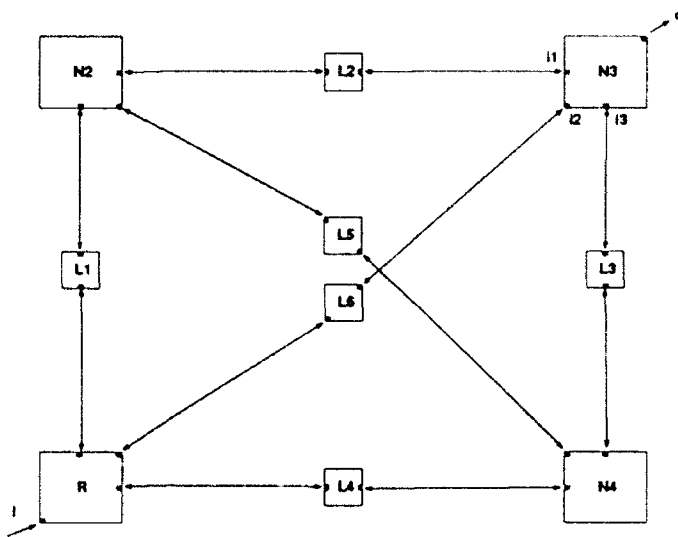


Figure 7 4 Node Network

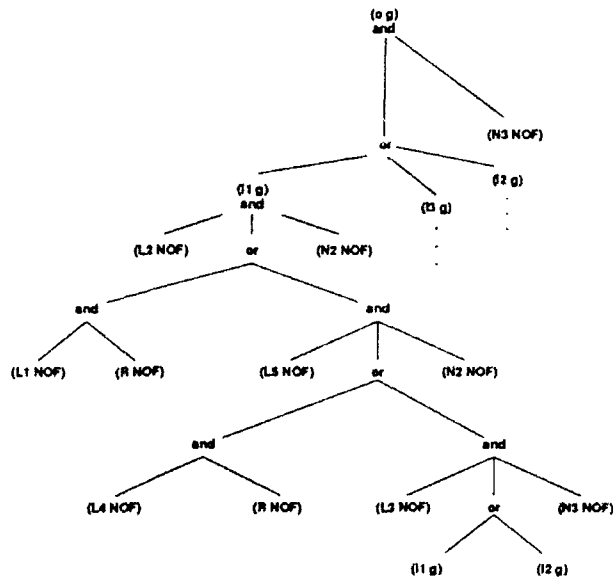


Figure 8 Tree with a Cycle

tracing will lead to an infinite tree. When this occurs then the tracing on the branch stops and all OR branches of the tree up to the first AND are trimmed. Thus, since (i1 g) is true at the top of the tree, "(i1 g) or (i2 g)" can be trimmed. If a condition is contradicted then RMG also stops tracing and trims all AND branches off the tree until the first OR.

FTP Figure 9 is a fault tolerant processor FTP with 4 channels (CH) and 6 network interfaces (NI). In the model for FTP, the NIs are subcomponents of the channels and the

channels are subcomponents of FTP. The output characteristics for the output ports from FTP are determined by the OCDs for the lowest level components NI. FTP is initially in mode UP. Some CHs and NIs are in a mode that means they are communication links to a network (not shown). Channel output is voted. Failure of a CH or NI may cause FTP to change to a recovery mode (RECOV). FTP then recovers via a transition back to UP, changes modes of failed CHs to non-operational (NOOP) and designates new CHs and NIs as communication links. Output from non-operational channels becomes "n" so they are no longer used in the vote.

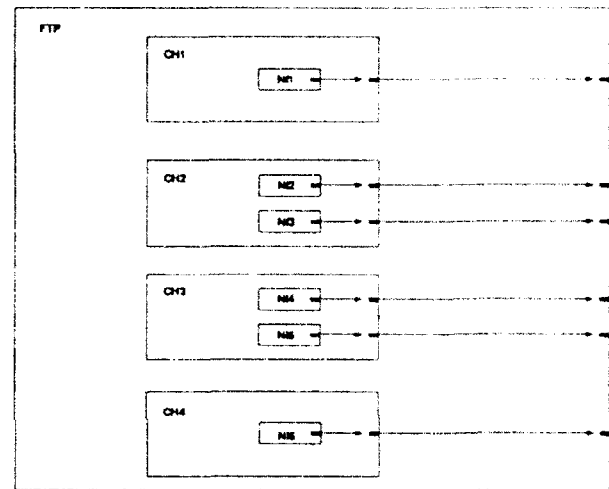


Figure 9 FTP

Network The network, NET-1, is shown in Figure 10. NET-1 has root nodes (R), nodes (N) and links (L) as subcomponents. Outside NET-1 are devices (DA, DB, DC and DD) and device interface units (DIU). NET-1 has three modes, UP, FAIL, and RECOV. Outputs from NET-1 to devices are "g" if the network is UP and the nodes connected to the outputs are not failed (NOF). They are "n" if the network is recovering (RECOV) and "b" if the network has failed (FAIL) or if the network is UP and the node connected to the output has failed. This output behavior is given in the node OCDs. The node OCDs depend on the NET-1 modes.

When NET-1 is in mode RECOV it tries to make a recovery transition that renders enough links "in-use" so there is a path from the root node to every other node. The network fails if there are not enough links or nodes to recover. The recovery transitions are an exhaustive list of initial states of NET-1, the nodes and the links and the inputs which give resulting "regrow" configuration which sets some links as in use and some as not in use.

The function of the network is to transmit information from devices. Devices of each type are voted and the system

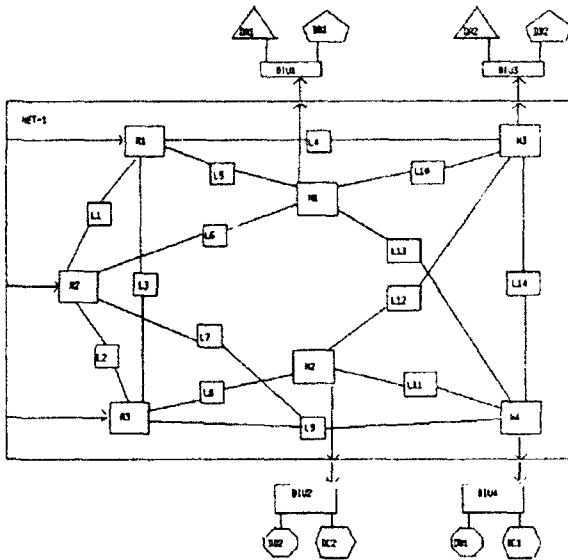


Figure 10 IAPSA Network

fails if a vote cannot be achieved within each group of device types.

Conclusions

The examples presented in this report were test cases providing experience in modeling with RMG. From this experience we were able to formulate conclusions about the RMG approach to modeling, the current capabilities of RMG and future changes or additions to RMG. The conclusions are presented in this section. They are explained in more detail in [3]

The tools for modeling components in RMG, *i.e.*, LRMs, OCDs, modes and transitions are adequate for modeling simple components. These tools coupled with RMG's graphical representation of a system with component pictures and connections provide a simple way for the user to specify effects of component changes on other components and on the system. The aggregation process performs a useful failure mode effects analysis (FMEA). The examples from the IAPSA architecture demonstrate that an algorithm for performing such a FMEA is feasible for complex systems.

The graphical representation is limited in that it does not display modes, transitions, and transition effects in a way immediately accessible to the user. A detailed specification of complex components such as FTP and NET models with the basic modeling tools and the graphical representation is impractical. Specification of these models is facilitated by hierarchical modeling, *i.e.*, creation of parent- and sub-components, and use of transition effects to model complex dependencies. Logical expressions resulting from the aggregation process can be prohibitively large.

The RMG model for the IAPSA architecture in [4] demonstrates the feasibility of a building blocks approach to reliability modeling. It was created in three stages. A model of the fault tolerant processor (FTP) was created first. A model of a single network (NET) was created second. One FTP and two NETs were copied into a new knowledge base and connected. System level transition effects were added. System failure conditions were added for generation of an ASSIST model.

The regrow model for the network in the IAPSA architecture revealed limitations of the language RMG uses to express OCDs and transitions. Over 2000 of these complicated transitions had to be generated outside of RMG. A proposal for remedy of this situation is enrichment of the ASSIST language to allow the user to create functions that compute complicated transitions such as network regrow.

Acknowledgements

The author would like to acknowledge W.C. Lee and C.M. McCann who wrote the initial specification for RMG. I also acknowledge the support and helpful comments of D.L. Palumbo and S.C. Johnson at AIRLAB at NASA-Langley.

References

- [1] Bavuso, S.J., P.L. Peterson and D.M. Rose, "CARE III Model Overview and User's Guide", NASA TM-85810, June 1984.
- [2] Butler, Ricky W., "The SURE Reliability Analysis Program", NASA TM-87593, 1986.
- [3] Cohen, G.C., P.R. Breckel, and A.J. Booker, "Reliability Model Generator", NASA Contractor Report, December, 1990.
- [4] Cohen, G.C., C.W. Lee and M.J. Strickland, "Design of an Integrated Airframe/Propulsion Control System Architecture", NASA Contractor Report 182007, March, 1990.
- [5] Cohen, G.C., M.C. McCann, "Reliability Model Generator Specification", NASA Contractor Report 182009, March, 1990.
- [6] Dugan, J.B., K.S. Trivedi, M.K. Smotherman and R.M. Geist, "The Hybrid Automated Reliability Predictor", Duke University, 1985.
- [7] Johnson, Sally C., "ASSIST User's Manual", NASA Technical Memorandum 87735, August 1986
- [8] Palumbo, D.L. and D.M. Nicol, "Generation and Analysis of Large Reliability Models", Ninth Digital Avionics Systems Conference, pages 350- 354, October 1990.

Computer-Aided Immunization and Opportunities for Data Visualization

Giles L. Crane
NJ State Department of Health
Division of Epidemiology
Hamilton, NJ 08625

Abstract

The importance of childhood immunization in NJ has become apparent once again because of the resurgence of measles in NJ. A National Health Goal has been set to bring about immunization against childhood diseases of diphtheria, pertussis, tetanus, infantile paralysis (polio), measles, mumps, and rubella by the age of two, rather than be the age of school entry. Furthermore, Congress has acted to encourage the manufacture of vaccines and practice of immunization by controlling litigation [5].

As part of a NJ task force effort to initiate early immunization in hard-to-reach inner city areas, staff designed, wrote, and debugged a program for registering, evaluating, and scheduling immunizations according to age appropriate immunization rules. Part of the work is on the interface of statistics and computer science, and also offers many opportunities for data visualization. This paper will describe this work, point out essential references, and present opportunities and directions both for data visualization and for increasing immunization coverage.

1. Attempts at a Computer Aided System

Initially, program staff looked for an established software and hardware systems. Among these was a WIC (Women, Infants, and Children Program) related package written in DBASE III+, a stand-alone program in RBASE written as a prototype, and a user interface written in DBASE IV. All three proved to be false starts, though each contributed insight to the present system. The DBASE III+ system appeared to be too integrated into an out-of-state WIC program system for easy adaptation to NJ requirements; the RBASE system was admirable in logical structure, but appeared to be too slow for the interactive situation at the two immunization stations; thirdly, the DBASE IV interface appeared to be costly and difficult in implementation, although perhaps offering more general programming facilities than EPI-INFO Version 5.0 [10].

There was also considerable experience within the NJ Health Dept. with EPI-INFO Version 5.0, micro-computer software developed jointly by cooperating small groups in the CDC and WHO, and currently supplied to EIS (Epidemiology Intelligence Officers of the Public Health Service). While there are serious faults with the underlying parsing and memory management algorithms of the current version (Version 5.01b) of EPI-INFO, the package appeared to draw upon very specific features relevant to Epidemiological Surveillance from many existing software (BASIC, DBASE, SAS), to have "immediacy" in data entry and validation techniques, to have low cost in case many more stations were necessary, and to have support for improvement in future versions.

An additional source of inspiration was the Johns Hopkins project to provide Haemophilus Influenza Type B vaccination to Navajo Infants via 11 remotely located immunization stations [8]. Shortage of NJ funds did not permit detailed consultation with personnel supporting this project.

2. Three Sets of Vaccination Rules

In order to write data editing, date projection, and evaluation programs, age appropriate vaccination rules for DPT, Td (adult tetanus and diphtheria), polio, hemophilus influenza, measles-mumps-rubella, and single dose measles, mumps or rubella from three medical authorities had to be reconciled. The three medical authorities were the American Academy of Pediatrics [1], the CDC Immunization Practices Advisory Committee (ACIP) [6], and the NJ Immunization Program [7] within the Div. of Epidemiology. No graphical display of any of these schedules were available. A future goal may be to visualize the schedules for individuals and populations, perhaps with reference to the Parisian train schedules popularized by Edward Tufte [11]. The rules differed slightly in numbers of shots required, minimum intervals, and various contingencies related to cutoff times.

Obtaining information on the rules from personnel in on-going immunization programs, proved to be more time consuming and demanding of patience than ex-

pected. The Immunization Initiative, as a start-up program, required very active participation by the Coordinator in order to establish the effort among other projects competing for resources at the Health Dept.

A fourth source of vaccination rules is the WHO Expanded Program for Immunization (EPI) [4]. In order to provide an effective program in a large bureaucracy, the World Health Organization has simplified vaccination requirements and the list of contraindications to a schedule much more basic than that in current NJ use.

3. Example of Programming

Exhibit 1 below is an example segment of the programming for data entry and checking of Polio shots, written in the EPI-INFO check language. While very "close", in a software sense, to the entry process of the date of the first Polio shot (OPV), the available programming structure is very limited (virtually just if-statements). Facilities to undertake complex computations are also limited, although there is a facility to call a subroutine written in Pascal. Date handling, including missing dates, is present, but memory variables (in distinction to variables included as data items in the dataset) are confined to real numbers.

4. Microcomputer Hardware

To start this initiative quickly, the Initiative Task Force borrowed a 286 laptop with modem and a small dot-matrix printer for one site, and a 286 desktop (no modem) and a wide carriage, dot-matrix printer for the second site. Development time was available most days on an IBM PS/2 386 tower with a modem and wide carriage, dot-matrix printer located at the NJ State Health Dept. The software which is now in use can be operated on an XT microcomputer with 640K of main memory.

To continue the initiative and the concept of a moveable, immunization team, the task force selected and ordered two notebook micro computers with modems and small (under 10 lbs) dot-matrix, narrow carriage printers. A third notebook system as a backup is under consideration. The microcomputers are powerful enough for much future enhancement to the software and hardware (an ethernet enhancement is available). At present, it is felt that, probably, the Initiative teams will operate in offices rather in a van with daily changes in location. Nevertheless, the ordered hardware includes the possibility. The task force also ordered a desktop for archiving, program administration, and survey use.

The two sites may be seen as the beginning of a statistically designed improvement experiment. Important factors included in the design are listed below:

Thus the demonstration phase of this project might be viewed as a 2×5 factorial design [11]. Other important factors which might lend themselves to systematic evaluation are hours, length of interview, and incentives. Other states with more station sites might have the opportunity for evaluating these factors and their interactions. Since there were only two stations available in NJ, we could not set up an improvement design, and had to content ourselves, at least initially, with looking for large negative effects or large positive effects.

5. Simple Computer-Aided Immunization

Exhibit 2 presents both the two screens necessary for entry and review of demographic and immunization records, and the page as printed for the child's parent or sponsor. The top half of the form (demographic information) is a screen and can be printed separately from the bottom (immunization record) screen of the form.

In a simple situation, a parent brings a child to the immunization station within the WIC clinic, fills out the demographic information in the first screen, and uses the paper immunization record (yellow card) to fill out the second screen. The evaluation and due dates are shown immediately, and are printed, so that the parent can take one copy with her to the nurse who will perform any vaccinations due. The clerk and nurse follow up children and parents to encourage age appropriate immunization. There is some opportunity to evaluate the general vaccination status of difficult to reach, inner city populations. In the brief setup time before the clinic opens in the morning, Initiative staff bring out the microcomputer and printer, which are kept in secure areas at night and on weekends. At noon and at the end of the clinic day, staff backup all datasets.

Various types of immunization records are encountered: NJ Immunization Record (yellow card), U.S. Military Immunization cards, and foreign private, national, and military records. Currently there is no provision to record unconventional vaccines (e.g. IGG, Hepatitis B, etc.) or exceptional characteristics of previous immunizations.

6. The Immunization Initiative After 3 Months

After operation from December, 1991 until February 1992, the immunization station in the inner city in Northern NJ has 1423 enrollees, is enrolling approximately 500 new children per month, of which 30% come into the station from outside the WIC program in order

Exhibit 1.

```

* WICIMM Version 1.0: EXAMPLE OF CHECK LANGUAGE: POLIO DOSE 1
OPV1
RANGE "01/01/40" "01/01/99"
type "*" -1 +0 48
enter
type " " -1 +0 48
if OPV1 < BIRTHDATE then
    HELP "SHOT DATE BEFORE BIRTHDATE !\nPLEASE CHECK SHOT DATE." 24 12
    CLEAR OPV1
    GOTO OPV1
endif
if OPV1 > today then
    HELP "SHOT DATE IN FUTURE !\nPLEASE CHECK SHOT DATE." 24 12
    CLEAR OPV1
    GOTO OPV1
endif
if OPV1 = DOB then
    CLEAR XOPV1
    CLEAR OPV1
    GOTO OPV1
endif
if OPV1 < BIRTHDATE + in1mos then
    XOPV1 = OPV1
    CLEAR OPV1
endif
if OPV1 = . then
    n1 = in2mos
    if XOPV1 - XOPV1 = 0 then
        n1 = XOPV1 + in2mos
    endif
    if agetdays >= in33mos then
        n1 = in33mos
    endif
    if agetdays >= in7yr then
        n1 = in7yr
    endif
    if n1 < agetdays then
        n1 = agetdays
    endif
    OPV1due = birthdate + n1
else
    CLEAR OPV1due
endif
UPDATE
END

```

FACTOR AFFECTING OPERATION	FACTOR LEVELS
(1) Latitude of clinic	North, south.
(2) Minority population	Afro-American, Hispanic.
(3) Computer	Desktop, laptop.
(4) Printer	Large dot-matrix with pull tractor, small dot-matrix with sprocket platten.
(5) Immunization site	With computer station at WIC, Not with computer but at health center.

to obtain immunization. In the southern inner city, operation for 2 months from January until February 1992, 916 children have been enrolled, about 450 new children per month are enrolling, 24% of whom were from outside the WIC program.

In both inner cities, enrollment appears to be concentrated from a relatively small area: most clients are from 5 zipcode areas in the northern inner city, and from 4 zipcode areas in the southern inner city.

A preliminary evaluation of need revealed that shots were overdue for 52% of children in the northern site and 46% of children at the southern site. The target populations are approximately 10,000 children at one site and 7,000 children at the other. (Initial enrollment data will be used to indicate the service area, and to improve the estimates of the target populations.)

7. Sources of NJ Health Statistics

Routinely collected health statistics also offer many opportunities for envisioning information. NJ health statistics are available from many diverse sources, most of which can be obtained in computer-readable form. Principal among these are birth tapes, death tapes (including Multiple Cause of Death tapes), Hospital Discharge tapes (MIDES or UB82), and the Cancer Registry. Other sources include drug treatment discharges, birth defects registry, medical claims tapes, the AIDS registry, communicable disease reports, sexually transmitted disease (STD) reports, TB cases, and sexual assault counseling forms. Still other sources within State government are fetal death tapes, poisoning reports, cervical cancer screening reports, fatal accident reports, homicide reports, family planning forms, hemophilia financial assistance data, mental retardation services, and community mental health center reports. Quality, completeness, and data gathering plan vary widely: some are required by law to be complete, some are voluntary, and many are accumulative in nature.

Childhood morbidity from vector borne diseases is still present, for example malaria, long associated with

the female *Anopheles* mosquito [2], as can be seen in the crude disease rates given below in Exhibit 3.

8. Current Statistics of Interest

Statistics of scientific interest at present are the coverage at 24 months (percent of babies completing vaccination by the age of two years), and coverage curves (percent of children competing age appropriate shots at various ages from 2 months to 5 years). Coverage will be explored by various racial groups, locations, WIC program participation, and other factors of interest.

Statistics of programmatic and administrative include Need (percent of babies requiring shots at entry into the Initiative), and Impact (percent of children gaining immunization protection as a result of the Immunization Initiative. Thus, envisioning information for the Immunization Initiative may fall into two broad areas: program administration and epidemiology, wherein we are concerned with identifying the populations and locations of low coverage and problems which can be addressed.

9. General Principles

At the 150th Anniversary of the American Statistical Association, held in Washington, D.C., 1990, J.W. Tukey reminded us of four cornerstones of statistical practice: First, to give estimates of mean values; second, to estimate variation; third, to indicate the multiplicity of possible effects; and fourth, to explicate possible biases which might affect results.

Analogously, in envisioning information, it would be basic to display mean values; show possible variation by confidence limits, by sets of realizable curves, or by some other technique; indicate multiplicity of effects by many views under various assumptions; and fourth, to indicate possible biases. I believe that the indication of possible bias (both direction and magnitude) has been neglected in many past and current attempts to envision information.

For example, in communicable disease work in the

Exhibit 2.

```

=====DEMOGRAPHIC RECORD SCREEN=====
WIC/SURVEY ID No. _____ W _ SITE _ INITIAL DATE __/__/__
CLIENT:LAST _____,FIRST _____ MI _ NO.VISITS __
STREET _____
CITY _____, STATE __ ZIP _____
PHONE _____ DEM. CHG. _
BIRTHDATE __/__/__ SEX _ ETHNICITY _ LAST DEM.CHG. __/__/__

MOTHER:LAST _____ M.FIRST _____ M.AGE:##
FULL TIME RESPONSIBILITY FOR CLIENT _

HOUSEHOLD:NO. CHILDREN ## UNDER 5 #

AFDC: RECIPIENT(Y/N) _ AFDC NUMBER _____ ATTEND DAYCARE _

SOURCE OF HEALTH CARE: 1=W, 2=S, 3=Imm.,4=WS.,5=WI,6=SI, 7=All, 8=NONE,9=UNK
LOCAL PUBLIC HEALTH _ COMMUNITY CLINIC _ HOSPITAL CLINIC _
EMERGENCY ROOM _ PRIVATE DOCTOR _ OTHER(SPECIFY)_

HEALTH INSURANCE(CODE) _ PAYS FOR WELL CHILD CARE (Y/N) _
BARRIERS (CODE) _ EXEMPT _ IMM.RECORD INIT.(Y/N) _ AVAIL _
=====IMMUNIZATION RECORD SCREEN=====
CH_ID _____ L _____ F _____ DOB__/__/__ AGE-DAYS####
DPT/DT DPT1 __/__/__ DT1 _ DPT1 Due __/__/__ XDPT1 __/__/__
DPT2 __/__/__ DT2 _ DPT2 Due __/__/__ XDPT2 __/__/__
DPT3 __/__/__ DT3 _ DPT3 Due __/__/__ XDPT3 __/__/__
DPT4 __/__/__ DT4 _ DPT4 Due __/__/__ XDPT4 __/__/__
DPT5 __/__/__ DT5 _ DPT5 Due __/__/__ XDPT5 __/__/__
Td TD1 __/__/__ TD1 Due __/__/__ XTD1 __/__/__
TD2 __/__/__ TD2 Due __/__/__ XTD2 __/__/__
TD3 __/__/__ TD3 Due __/__/__ XTD3 __/__/__
Polio OPV1 __/__/__ OPV1 Due __/__/__ XOPV1 __/__/__
OPV2 __/__/__ OPV2 Due __/__/__ XOPV2 __/__/__
OPV3 __/__/__ OPV3 Due __/__/__ XOPV3 __/__/__
OPV4 __/__/__ OPV4 Due __/__/__ XOPV4 __/__/__
HIB HIB1 __/__/__ HIB1 Due __/__/__ XHIB1 __/__/__
HIB2 __/__/__ HIB2 Due __/__/__ XHIB2 __/__/__
HIB3 __/__/__ HIB3 Due __/__/__ XHIB3 __/__/__
HIB4 __/__/__ HIB4 Due __/__/__ XHIB4 __/__/__
MMR MR1 __/__/__ M1 _ MR1 Due __/__/__ XMR1 __/__/__
MR2 __/__/__ M2 _ MR2 Due __/__/__ XMR2 __/__/__
MR3 __/__/__ M3 _ MR3 Due __/__/__ XMR3 __/__/__
Me __/__/__ Mu __/__/__ Ru __/__/__
PPD _ IMM. CHG __/__/__ TODAY __/__/__ EVALUATED BY _
=====

```

Exhibit 3.

NJ MALARIA MORBIDITY

YEAR	Number	Rate/ 100,000
1974	12	0.16
1975	15	0.20
1976	17	0.23
1977	19	0.26
1978	30	0.41
1979	19	0.26
1980	66	0.90
1981	50	0.67
1982	39	0.53
1983	29	0.39
1984	41	0.55
1985	19	0.25
1986	36	0.47
1987	41	0.53
1988	59	0.76
1989	62	0.80
1990	81	1.05
1991	61	0.79

Public Health Departments, bar charts and line charts are very common indeed. They are certainly used to display the numbers of confirmed cases of a specific disease found in surveys, and sometimes varieties of confidence limits based on sampling or yearly variation are envisioned. Multiplicity of effects (age, race, sex, location, other subpopulations) are often shown in multiple charts. However, bias, even such matters as reporting delay, possible selection bias, etc. may not be indicated graphically. The result may be a less than satisfactory feeling for the actual numbers of cases which may exist, and consequently, an unjustified feeling of security by officials who see just reported cases displayed. Clearly, improvements in envisioning possible bias, in data in many fields as well as in Public Health data, can be made.

10. Future Directions

Besides conventional data visualizations (scatter diagrams, histograms, graphs), envisioning individual age appropriate schedules as well as population results may aid in popularizing the project, helping mothers to recall shot schedules, and other program evaluation charts to provide visual guidance. Hepatitis B vaccination may be required in future (Singapore, for example, requires

universal Hepatitis B vaccination).

Outreach activities may involve a risk analysis of dropping out; this will be very useful in guiding outreach activities. For outreach, labels can be produced, as well as lists of those requiring future shots for volunteer workers within the communities, or perhaps zipcode areas. Techniques involving autodialing, autodialing in combination with voice as well as reminder cards may be attempted to reach those with telephones. It may also be possible to initiate enroll starting with birth certificates rather at WIC or other agency sites, providing mothers, or whoever is sponsoring the child, can be persuaded of the importance of immunization.

Further medical input as well as information on the health condition of the child in order to provide the optimum immunization FOR THIS PARTICULAR CHILD. One large step in this direction would be to implement the list of contraindications.

There have been suggestions by authors in the medical literature as well as in various programs to establish a State or National Vaccination Registry, an enormous undertaking. On a smaller scale, the present Immunization Initiative will be the object of a program evaluation study, and perhaps other studies aimed at increasing coverage and early immunization. At present we are survey-

ing the experience of other states. At the time of writing, other members of our task force have concluded that our computer-aided registration and assessment system is state-of-the-art.

11. Acknowledgements

The principle contributors to the NJ Health Dept. Immunization Initiative are Bonnie Schuster, coordinator, grant writer, and dedicated public health professional, Kenneth J. Bryd, Public Health Representative.

The contributions of the following people are gratefully acknowledged: Lyn Finelli composed the original survey form which included demographic information and vaccine dose information. Igor Bulim of the CDC Immunization Section programmed a prototype software system in the RBASE language which influenced development of the projection and evaluation sections. Ken O'Dowd contributed the original form feed and print screen utilities. Ken J. Bryd and James Lutz of the NJ Epidemiology Division composed and evaluated two sets of example forms. Deborah Jones and Agnes Phares of NJ WIC loaned equipment. Ken Spitalny answered medical questions pertaining to the consolidation of the three sets of immunization rules, as did Charles O'Donnell, Coordinator of the NJ Immunization Program. Harvey of the Health Dept. MIS group generated several appropriate ideas concerning the items on the survey form, and evaluated DBASE IV as a possible package to be used for this program. Luke Hilgendorff of the NJ Health Dept. MIS group also aided in the evaluation of database packages and provided useful guidance.

12. Disclaimer

The contents of this paper should in no way be interpreted as representing the policy of the New Jersey Department of Health. The use of brand names in this article is for purposes of identification and does not imply any endorsement for the mentioned products by the New Jersey Department of Health.

13. References

- 1 G. Peter, MD, ML Lepow, MD, GH McCracken, Jr., MD and CF Phillips, MD, editors (1991). Report of the Committee on Infectious Diseases, Twenty-second Edition, 1991. American Academy of Pediatrics, 141 Northwest Point Blvd., P.O. Box 927, Elk Grove Village, Illinois 60009-0927. "The Red Book".
- 2 Abram S. Benenson, edit. (1990). Control of Communicable Diseases in Man. American Public Health Association, 1015 Fifteenth Street, NW, Washington, DC. 20005
- 4 — (1991). Plan to Provide Immunization Services. Expanded Programme on Immunization, World Health Organization, 1211 Geneva 27, Switzerland.
- 5 National Childhood Vaccine Injury Act of 1986.
- 6 Recommendations of the Immunization Practices Advisory Committee. U.S. Dept. of Health and Human Services, Public Health Service, Centers for Disease Control (CDC), Center for Infectious Diseases, Atlanta Georgia 30333. (Publishing vehicle is MMWR.)
- 7 — (1992). Guidelines for Implementing New Jersey's Immunization Requirements for Pupils in School: A Practical Guide for School and Health Officials in Implementing Chapter 14 of the NJ Sanitary Code. NJ Dept. of Health, Div. of Epidemiology, Immunization Program, Trenton, NJ 08625-0369. (1992 edition in press.)
- 8 M. Santosham, Mark Wolfe, et al. (1991). The Efficacy in Navajo Infants of a Conjugate Vaccine consisting of Haemophilus Influenza, Type b Polysaccharide and Neisseria Meningitidis Outer Membrane Protein Complex. New England J. of Medicine, June 20, 324:25, 1767-1772.
- 9 F. MacFarlane Burnet (1969). Genes, Dreams, and Reality.
- 10 EPI-INFO Version 5 (1991). Epidemiology Program Office, Mailstop G34, Centers for Disease Control, Atlanta, GA 30333.
- 11 Edward Tufte (1991). Envisioning Information. Graphics Press, Box 43, Cheshire, Connecticut 06410.
- 12 Box, George E.P., Hunter, William G., and Hunter, J. Stuart (1978). Statistics for Experimenters. John Wiley & Sons, New York. Chapter 10.

Enhancing Tactical Direct Fire Synchronization Measures

by

David A. Dryer
TRAC-MTRY
P.O. Box 8692, Naval Postgraduate School
Monterey, CA 93943-0692

Harold Larson, William Kemple, Robert W. Lamont
Department of Operations Research
Naval Postgraduate School
Monterey, CA 93943

ABSTRACT

This paper develops objective U.S. Army doctrinal measures of Direct Fire Synchronization of Brigade, Battalion, and lower level units. Graphical portrayal of these measures, using emerging Visual Data Analysis techniques including 3D, color, and animation of small multiples is demonstrated. Such objective measures are of use in assessing the current capabilities of Army units employing new tactical concepts. These graphical displays will provide enhanced capabilities in meeting current and projected combat simulation analyses of weapon characteristics, unit mission effectiveness, and battlefield operating system measures.

PROJECT CONCEPT

Objective: BEAM will develop objective measures of tactical doctrinal tenets to resolve deficiencies in data collection requirements at Combat Training Centers and improved identification of key performance weaknesses in the use of tactical doctrine.

Approach: Definitions of doctrinal tenets of tactical forces will be solicited from doctrine and subject matter experts, including personnel at the Command and General Staff College and the National Training Center (NTC). All definitions received from these sources will be considered in terms of available measurable quantities, from NTC battles and Janus(A) combat model simulation of such battles, with the goal of deriving objective measures of performance reflecting the selected tenets for tactical units. The framework for the presentation of these visual display indicators will be investigated in terms of user interface and system requirements. Selected visual displays will be prototyped. Should other measures seem appropriate, requiring data not currently available from NTC or the Janus(A) combat model, these will be described with appropriate indications of additional data needed for their evaluation. It is anticipated that

measures suggested may depend on the mission involved as well as the unit(s) employed in the accomplishment of that mission. To the extent possible, all suggested measures will be illustrated with actual observed data from NTC and/or the Janus(A) combat model. (See Figure 1.)

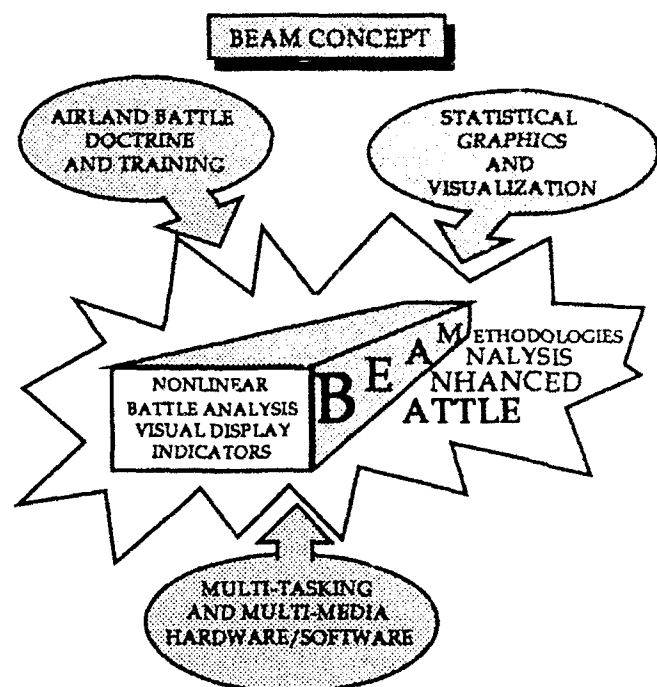


Figure 1

PROJECT OBJECTIVES

The main objective of the BEAM project is the development of computer displays which meaningfully picture synchronization at the tactical level. This in turn requires the identification of measurable

attributes which portray synchronization. Synchronization is a broad term and encompasses many different activities and processes, including coordination and integration of separate parts, both before and during the battle; indeed it is also meant to be a result, as stated in *Field Manual 100-5* (FM1005 [1]), with the goal of maximizing combat power at the decisive point. In FM 100-5, "combat power" is defined to have four components: Maneuver, Firepower, Protection and Leadership. Of these, FM 100-5 identifies Leadership as being the most essential. Unfortunately, as also pointed out in this manual, there are no ready formulas for measuring this essential element, so it has not been considered as a possible candidate for displays. The two components of Protection (making soldiers, systems, units difficult to locate and destroy, plus health and morale issues) are also not easy to quantify for display and have not been actively considered.

The remaining two components of combat power, Maneuver and Firepower, can be quantified in a number of ways and have been studied as candidates for useful computer displays of synchronization of a tactical force. The following sections provide discussions and illustrations of possible ways to usefully display these two components of combat power.

DOCTRINE REVIEW, EXPERT OPINIONS

A survey of current U. S. Army doctrine has been undertaken, to find the latest interpretations of the battlefield tenets: Initiative, Agility, Depth and Synchronization. The main focus of the initial effort is synchronization, which has many facets, some of which are more easily quantified and portrayed than others. *Field Manual 100-5* (FM 100-5 [1]) defines synchronization as follows: Synchronization is the arrangement of battlefield activities in time, space, and purpose to produce maximum relative combat power at the decisive point. Synchronization is both a process and a result. This manual stresses the importance of synchronization in both offensive and defensive operations. Indeed, it points out that a good defensive procedure is to interrupt the synchronization of the attacker.

In *Field Manual 71-2* (FM 71-2 [3]), synchronization is called one of the four characteristics of successful operations (the other three are the remaining tenets listed in FM 100-5). On page 1-6 it is referred to as "Synchronization is the process of integrating the activities on the battlefield to produce the desired result. Synchronization of operations is required in order to maximize the combat power of the combined arms team. It requires a command, control, and communications system that can mass and focus the combat power of the task force at the decisive time and place." In discussing synchronization of offensive

operations (page 3-28), it is stated that the commander and staff synchronize and integrate all combat, combat support and combat service support assets organic and available to the battalion task force. It does not indicate how this is to be achieved. In discussing synchronization of defensive operations, (page 4-24) it states that "The success of the defense is determined by how effectively all supporting organizations are integrated into the maneuver plan." A discussion of the sequence of the defense is given, but no reference is made to synchronization. References [2], [4]-[7] provide similar guidance.

In addition to guidance provided by these references, the Tactical Commander Development Course (TCDC) at Ft. Leavenworth, Kansas, provides tactical level training for unit commanders and staff personnel. The course stresses that effective synchronization requires many things, including:

- Anticipation of enemy actions
- Mastery of time-space relationships on the battlefield.
- Unity of purpose
- Understanding of weapons capabilities
- Knowledge of battlefield decision points.

This course is very useful in illustrating the complexities involved in tactical level planning and execution, and in highlighting the many facets of synchronization.

The National Training Center (NTC) at Fort Irwin, California, is the Army's primary tactical training location. This center features a highly instrumented one thousand square mile force-on-force training range. It employs laser devices to simulate engagements and their results, with a highly motivated Soviet-style force pitted against the visiting tank battalion. After Action Reviews (AARs) are employed (at many levels) following the battles fought by the visiting battalion to highlight correct use of current tenets, and to suggest possible areas of improvement. These AARs make use of the JANUS(A) combat model to portray the battle just fought, indicating position deployments, kills made and suffered, movements and timing, and many other things. The proposed BEAM displays are expected to improve the objective tools available to illustrate correct use of the battle tenets and highlight areas for improvement.

USER INTERFACE

The current candidate BEAM User Interface is pictured in Figure 2. This has been discussed with personnel at both the TCDC and the NTC for possible modification. As pictured there, the user would select one of the current tenets, the scenario type, the Battlefield Operating System (BOS) of interest, as well as which aspect of the tenet-BOS combination is of interest. This would then bring to the screen an

appropriate display for the combination selected. Several candidate displays have been designed, two of which are briefly mentioned below.

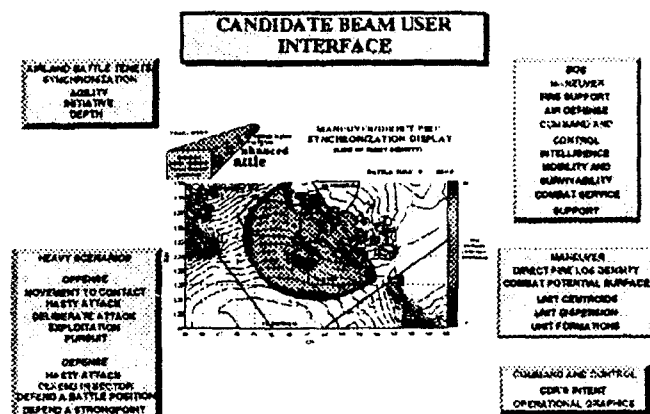


Figure 2

LINE OF SIGHT (LOS) DISPLAY

As discussed previously, a result of synchronization is the production of maximum relative combat power at the decisive point; one component of combat power is Firepower, the provision of destructive force essential to defeating the enemy's ability and will to fight. Direct fire weapons are a major component of this destructive force; the effectiveness of these weapons, in turn, depends on their lines of sight, the (possible) target areas they are able to see. If a force has placed its direct fire weapons in locations where they cannot see the main battle area, they will not be effective in contributing to the combat power in that area. Thus, the lines of sight for any particular placement of direct fire weapons control their effectiveness in any battle. If the lines of sight are massed at the decisive area of the battle (at the correct point in time) these weapons will be able to contribute to the combat power of the force; if lines of sight do not exist (at this time) they do not contribute to the combat power.

This line of reasoning leads to consideration of a graphical display of the lines of sight available to direct fire weapons, given their deployment positions (and realistically, their orientation at these positions). The JANUS combat model, within its own resolution, is able to determine the lines of sight between any two points within the area being depicted. This then allows one to build an LOS surface whose height at any point on the ground is given by the number of direct fire weapons, within the force, which are able to see that point; the higher the surface at a given point, the greater is the massing of lines of sights at that point. This surface can then be colored according to its height and displayed in two dimensions. The changes in

colors over a given area reflect the changes in the massing of the lines of sight, for the given disposition of the weapons used. A display of this type is given in Figure 3.

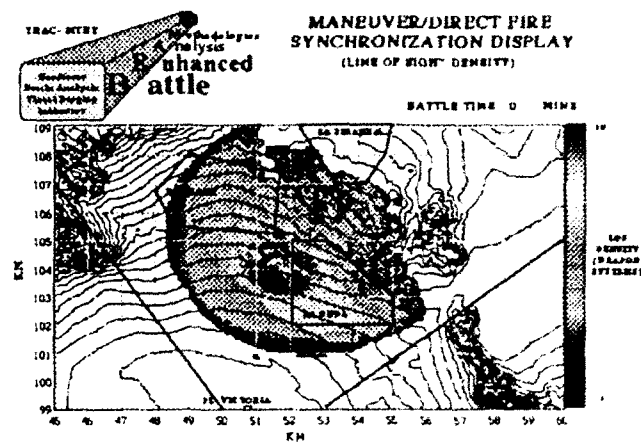


Figure 3

This figure portrays a portion of the NTC with 10 Blue tanks located in the top center of the screen in an area bounded by y-coordinates 105 and 107, x-coordinates 51 and 53. The number of tanks which can see any given point then varies from 0 to 10; since each tank has an effective maximum firing range, there is no purpose in portraying the existence of lines of sight beyond this range (in terms of the firepower of the defending tanks). Thus, the figure gives the number of these tanks which can see a given point, so long as that point is within the effective firing range of the tank. These numbers in turn are color coded with dark blue representing 1 tank having LOS; the colors range through lighter blue to green to yellow to red to signify the number of tanks with lines of sight varying up to 10. Thus, the massing of the lines of sight is given by the hot red color, with the cold blue color indicating a single tank is able to see the point. The lines of sight are aggregated only for points within the firing range of the given weapon.

This type of display gives a single-shot picture which is useful in portraying the massing (or lack of massing) of part of the combat power of the force at a given point in time. Constructing such snapshots at various points in time, and running them sequentially, can give a valuable indication of how the massing of lines of sight changed during the battle, due to either movement or attrition of the weapons portrayed. The opposing force locations can also be overlaid on the same picture, giving a more complete description of the dynamics of the battle. (See Figures 4, 5, 6 and 7.)

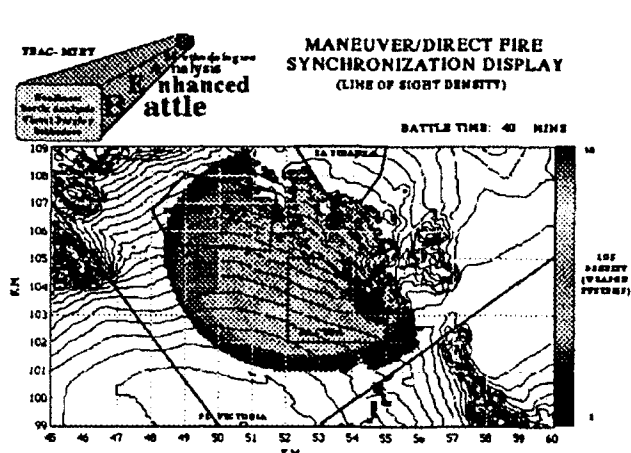


Figure 4

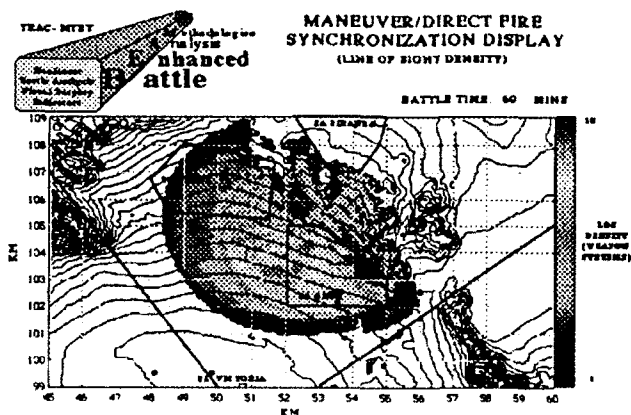


Figure 5

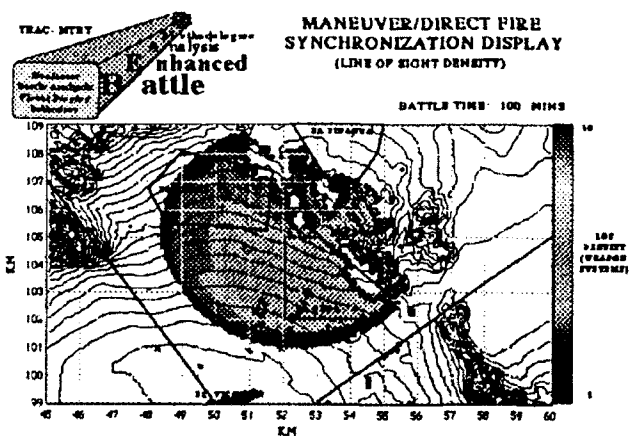


Figure 6

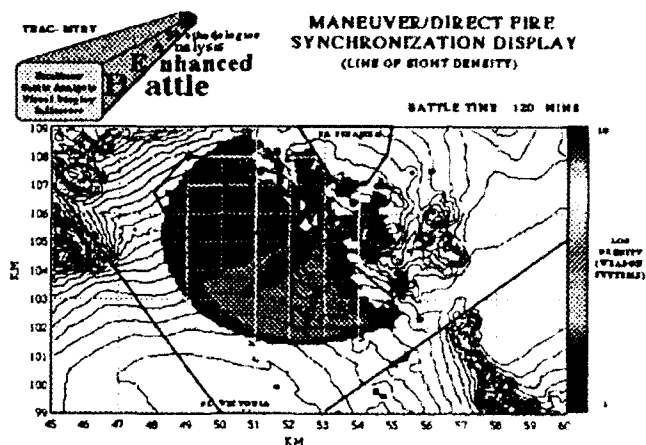


Figure 7

DESTRUCTION POTENTIAL (DP) DISPLAYS

Displays similar to the LOS displays described above can also be useful in providing additional information. The LOS display simply describes the absence or presence of line of sight between points, giving no indication of how much damage the weapons which have line of sight might be able to inflict on the corresponding point. There are a number of ways that a display can indicate the amount of destruction that could be inflicted at each point in a given area; this type of display is called a Destruction Potential display.

The DP display can be constructed in a number of different ways. Three different candidate DP displays are currently under study. One possible measure for a DP display is based on the expected number of kills, if every weapon available were to fire rounds at its maximum possible rate over a short period of time, say one minute; this measure is similar in spirit to the Quantified Judgment Model of [8].

Rather than basing a DP display on the expected number of kills, as above, a single-shot kill probability could be employed instead. That is, if all weapons which have LOS to a given point were to simultaneously fire one round at a target located there, it is a simple matter to compute the probability of scoring a kill at that point. This measure does not account for the rates of fire of the firing weapons, a quantity most would agree is an important contributor to the combat power of the force. To also bring this effect into play, the DP surface could be modified to represent the probability a target at the given point is killed, granted all weapons fired at maximum rate for a one-minute period. Which of these alternative methods of displaying a DP surface may be more useful for the Battalion Commander is currently under study.

In addition to these candidate DP displays, a number of possible displays illustrating maneuver are currently under development.

REFERENCES

- [1] *Operations*, FM 100-5, Commander, TRADOC, Fort Monroe, VA 23651-5000, 5 May 1986.
- [2] *Armored and Mechanized Infantry Brigade*, FM 71-3, USAARMC, ATTN: ATZK-DS, Fort Knox, KY 40121-5000, 11 May 1988.
- [3] *The Tank and Mechanized Infantry Battalion Task Force*, FM 71-2, Commandant, US Infantry School, ATTN: ATSH-B, Fort Benning, GA 31905-5410, September 1988.
- [4] *Mission Training Plan for the Tank and Mechanized Infantry Battalion Task Force* ARTEP 71-2-MTP, Commandant, US Army Infantry School, ATTN: ATSH-I-V-T-C, Fort Benning, GA 31905-5007, 3 October 1988.
- [5] *Tank and Mechanized Infantry Company Team*, FM 71-1, USAARMC, ATTN: ATZK-DS, Fort Knox, KY 40121-5000, 22 November 1988.
- [6] *Synchronization of Combat Power at the Task Force Level: Defining a Planning Methodology*, C. L. Long, Master of Military Art and Science, USA Command & General Staff College, ATTN: ATZL-GOP-SE, Fort Leavenworth, Kansas 66027-5070.
- [7] *Techniques and Procedures for Tactical Decisionmaking, Student Text* 100-9, US Army Command and General Staff College, Fort Leavenworth, Kansas, July, 1991.
- [8] *Understanding War, History and Theory of Combat*, T. N. Dupuy, Paragon House Publishers, New York, 1988.

EXPLORATION AND ANALYSIS OF DATA FROM A RCRA LAND TREATMENT FACILITY

by: Gary L. Patton, Elaine M. Armstrong,
Michael W. Holder, Katharine L. Coley, Robert C. Wallace
Radian Corporation, P.O. Box 201088, Austin, TX 78720-1088

ABSTRACT

Collection and chemical analysis of soil and soil-pore-liquid samples during a two year RCRA Land Treatment Demonstration (LTD) generated several megabytes of data to be used to determine the efficacy of a land treatment facility (LTF) in degrading, transforming, and immobilizing waste constituents within the treatment zone. A relational database implemented on a Unix workstation was used to manage and query data leading to data analysis. However, evolution of the experimental design left investigators with a large volume of data and little information. This paper presents the graphical and traditional statistical techniques used to determine constituent distribution throughout the LTF soil profile and identify possible mechanisms for constituent migration. Graphical techniques ranging from simple dot plots to use of the SAS INSIGHT exploratory data analysis tool will be presented along with the known problems each tool solved and the unsuspected problems each tool identified.

INTRODUCTION

A Land Treatment Demonstration (LTD) is conducted in accordance with regulations of the Resource Conservation and Recovery Act (RCRA) to determine the ability of a land treatment facility (LTF) to degrade, transform, or immobilize hazardous waste constituents applied to the land surface under a specified set of operating conditions. The criterion for determining the efficacy of the facility is presence or absence of waste constituents below the treatment zone. The presence of naturally occurring constituents (for example, inorganics like barium, manganese, etc) is defined as statistically greater concentrations of constituents below the LTF than in the background. The absence of these naturally occurring constituents is defined as concentrations below the LTF that are not statistically greater than in the background. Background concentrations are those found in the same soil series but in an area that has not been

affected by LTF or other waste management/disposal operations.

Figure 1 portrays a conceptual diagram of a land treatment facility. The treatment zone extends from the original land surface to a depth of 5 feet. The depth orientation of this diagram is exaggerated to emphasize the three dimensional (3D) aspects that were important during data analysis.

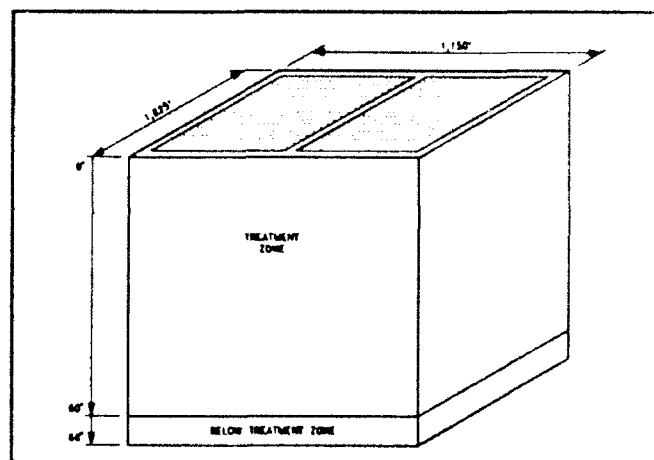


Figure 1.

The original sampling design for this LTD was established assuming that waste migration would be homogenous throughout the soil profile and quarterly sampling for two years would adequately characterize the entire soil profile. This design was based on EPA guidance for unsaturated zone monitoring. Since this sampling design was statistically simple, the experimental design called for a simple means comparison of LTF samples and background using the Student's t-test. However, there was a stipulation that other statistical methods could be proposed during data analysis if the Student's t-test was shown to be inappropriate.

The magnitude of the number of data points collected made data analysis more complex than had been expected, based on the simple sampling design. This paper presents some of the problems encountered during initial data analysis efforts and describes two graphical data visualization techniques that gleaned information from the mass of data collected.

DATA MANAGEMENT

A relational database was designed and implemented to manage data collected during the LTD. This database managed sample information, field data, analytical chemistry results of all samples, and quality control (QC) data for the analytical chemistry. Figure 2 presents a summary of the types of data managed during the LTD. In total approximately 25 megabytes (Mb) of data were collected and managed in the database.

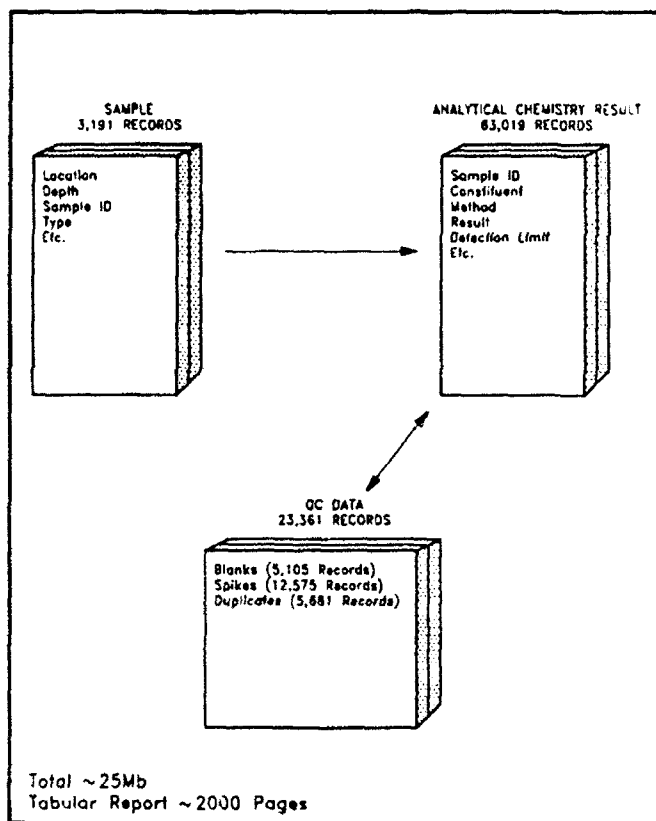


Figure 2.

The relational database package chosen for implementation of the data management plan used the Standard Query Language (SQL). It was originally planned that querying the database through SQL would enable us to identify and implement an appropriate

progression of data analysis methods. SQL is a very powerful tool and did enable us to query the database in fairly complex ways, but the tabular reports that contained output from queries were still too detailed to help the project team focus on priority issues for data analysis. In effect, we couldn't see the forest for the trees.

THE DATA ANALYSIS PROBLEM

The magnitude of data available for this LTD was almost as formidable as the cost of the project (~\$1.3 million); both emphasized the need for effective data analysis. This was not a project that could be redone. Therefore, graphics were used to summarize data and help lead and focus data analysis efforts. The rest of this paper describes two examples of how graphics were used to solve problems and focus data analysis efforts.

Examples of problems using tabular data reports

The two examples presented in this paper solve the following two problems encountered using tabular data reports:

- An unexplainable, apparent increase over time of one constituent in soil-pore liquids.
- Difficulty in visualizing the location of detectable concentrations below the treatment zone in relation to detectable concentrations within the treatment zone.

These two problems are described in more detail below with the graphical technique that was used to solve the problem.

First Example - Problem

The objective of the LTD was to analyze two environmental matrices (soil and soil-pore liquids) to determine the ability of the LTF to transform, degrade, or immobilize waste under a specified set of operating conditions. Query output tables of the detectable concentrations in the below treatment zone seemed to indicate that one constituent was increasing over time in soil-pore liquids. This was a concern because this constituent had never been detected in the waste that was being applied to the LTF soil surface.

First Example - Solution 1

The data visualization technique used to solve this problem involved graphing the sampling round (quarter)

against concentration detected in the samples; and label each point with a location identifier. Figure 3 presents an example of this graph. In Figure 3 it was apparent that there was no increase over time at any one location. While this resolved the problem of an increasing trend, it raised the problem of low-level concentrations that were fairly consistent in several of the soil-pore-liquid sampling locations. But now an engineer or geologist could focus their efforts on determining the source for this constituent instead of trying to identify a mechanism that would cause concentrations to increase.

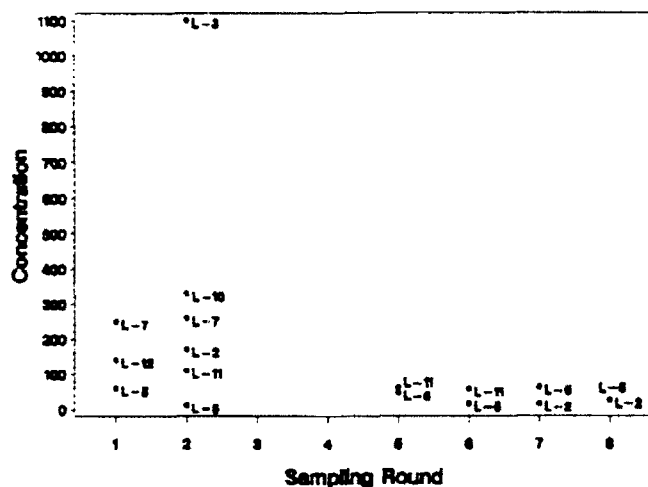


Figure 3.

Second Example - Problem

No one could reasonably expect to read the approximately 2000 page tabular report containing all results from the LTD and understand where waste constituents were detected in both treatment zone and below treatment zone soils; let alone prioritize their efforts to determine why constituents were detected. Therefore, determining the mechanism for possible constituent migration (if any were in fact migrating) posed an extremely difficult problem. To resolve the problem the project team decided to use SAS/INSIGHT's data exploration capabilities to glean information from the dataset.

Second Example - Solution 1

Initially, bar graphs (i.e., histograms) were created and reviewed. INSIGHT's interactive capabilities let the

project team focus on constituents that were a primary problem in the interval below the treatment zone. Figure 4 presents four bar graphs indicating the count of detectable concentrations in each depth of the LTF (4" depth was the interval monitored below the treatment zone); in each unit (the LTF was divided into two units); and for each constituent (a short list was used for this example). Note that the bars represent total number of detectable concentrations, not total number of samples. Figure 4 uses INSIGHT's highlighting capability to indicate which records in the unit and constituent graphs are associated with the 4" depth in the depth graph. INSIGHT is an interactive tool that allows the user to highlight portions of any graph and view the corresponding records in other graphs that are active on the screen. Of the 13 possible constituents, only "C", "D", "E", "G", "K", and "M" were detected in the depth below the treatment zone (see Figure 4). This kind of data visualization helped focus our analysis efforts.

Second Example - Solution 2

INSIGHT has a rotating 3D capability and it was thought this would help us to understand some of the spatial relationships of the data. However, the 3D scatter plot shown in Figure 5 was not expected. The planes of results were an artifact of the sampling plan. The sampling plan used compositing to collect 4 samples that would represent the 0-12", 12-36", 36-60", and 60-66" depths). Routinely, compositing is used to collect samples that represent a larger area, or volume, of soil than a discrete sample might. Compositing is similar to "averaging", some vertical variability information is lost, but knowledge of the average over an interval is gained. While this made economic sense, it did limit data analysis efforts due to the loss of some vertical information in the soil profile.

However, the rotating 3D capability could still be used to look across any depth to see if waste constituents were evenly distributed across the depth interval. Figure 6 presents a combination of bar graph and 3D scatter plot that was used to look at the spatial distribution of detectable concentrations. With labeling, it was possible to identify the constituents that were detected in each sample. After using Figure 6 to identify locations of frequently occurring constituents, a 3D scatterplot similar to Figure 7 could be used to look for concentration trends.

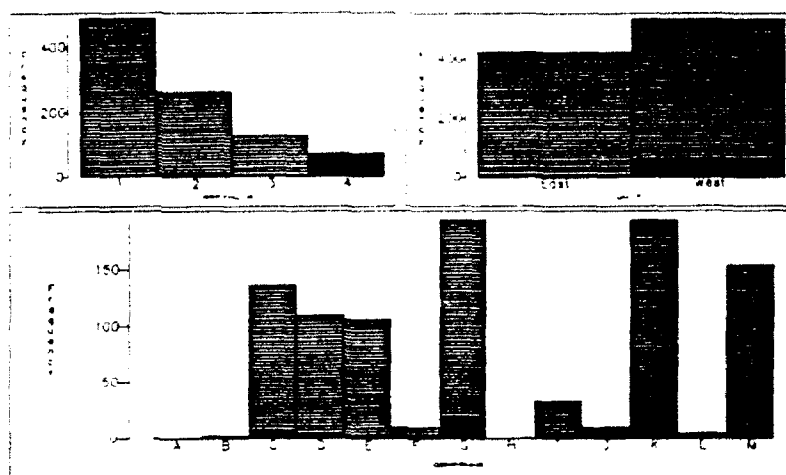


Figure 4.

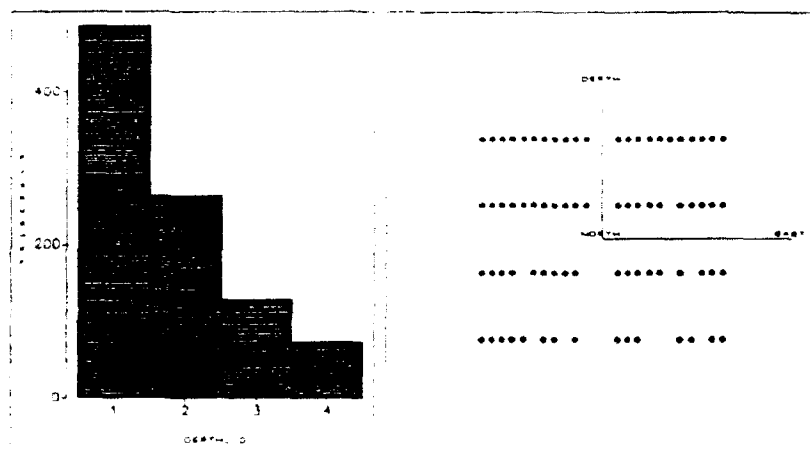


Figure 5.

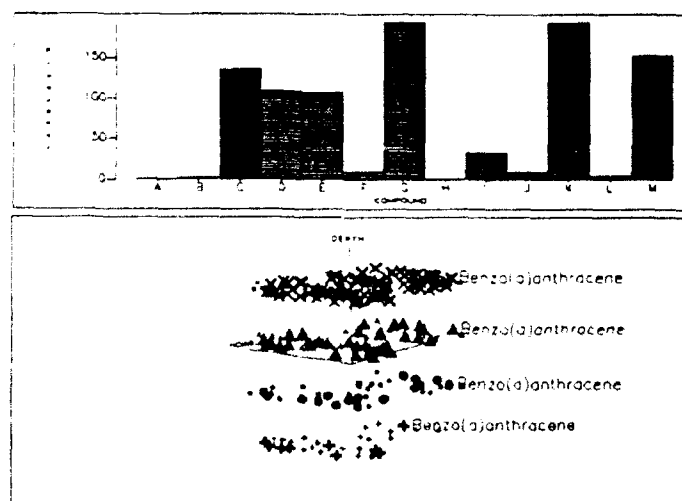


Figure 6.

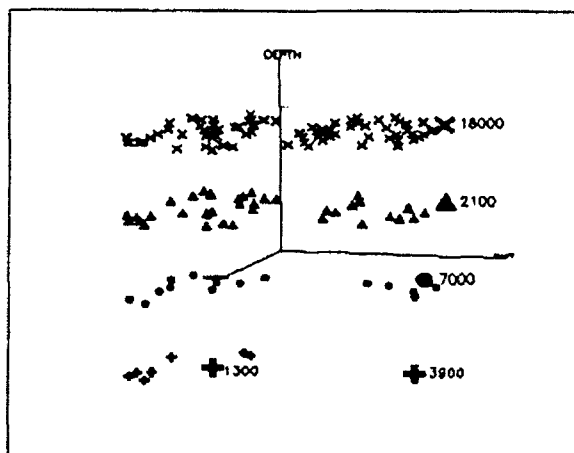


Figure 7.

A series of these kinds of 3D scatter plots and bar graphs helped to determine trends in detectable concentrations throughout a depth and throughout a vertical profile. Together, these scatter plots helped to focus data analysis and interpretation on priority problems such as:

- determining the kinds of constituents detected throughout the LTF;
- determining critical waste constituents;
- determining possible mechanisms for migration; and
- identifying appropriate operating conditions for the LTF.

Answers to "other" problems

Graphics helped throughout data analysis efforts and reporting to visualize information. Not only did the graphics help identify priorities and glean information from raw data, graphics also helped present both the data to be compared and the statistical decision tree used to determine an appropriate statistical comparison. Figure 8 presents a series of bar graphs that would be placed in a report to help indicate the distribution of detectable concentrations in both the LTF and background. These graphs helped demonstrate graphically when detectable concentrations were greater in the LTF than in background; a demonstration that is easier to interpret than a table of probability values indicating the statistical conclusions. These kinds of graphs also helped cognizant reviewers evaluate the statistical conclusions in light of practical considerations (i.e., a pictorial evaluation of the practical consequences of alpha and beta error). Figure 9 presents the decision tree that led to the statistical

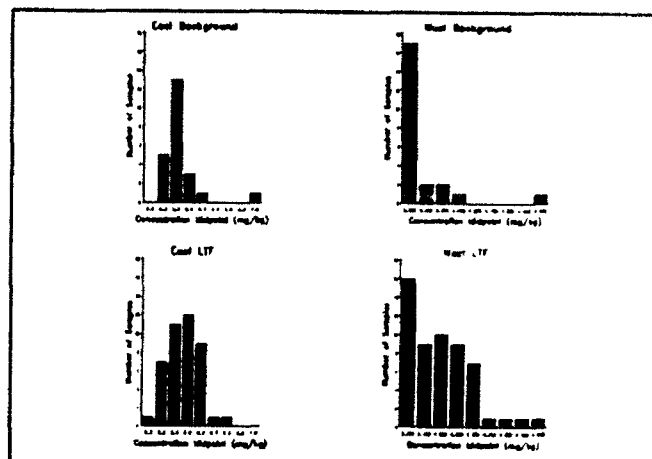


Figure 8.

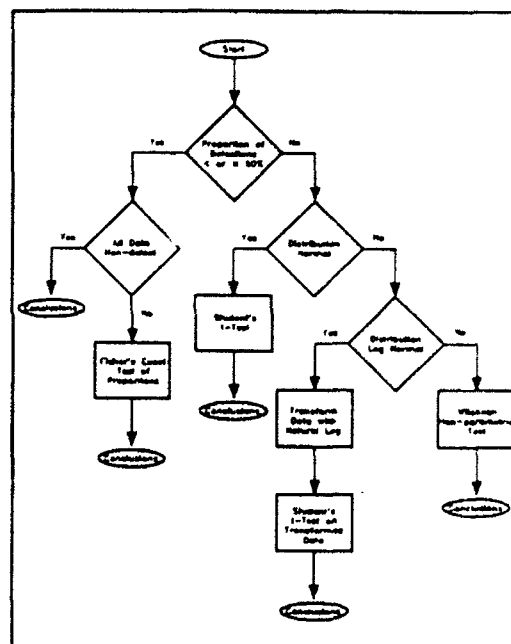


Figure 9.

methods that were used to compare LTF and background results, depending on the frequency of occurrence (detectable concentrations) and ability to meet statistical assumptions.

CONCLUSIONS

In conclusion, the old adage, "A picture is worth a thousand words, or tabular report pages." was found to be true. Graphical data analysis helped to identify priority issues and focus personnel and monetary resources.

MISHA – A Computational and Graphical Tool

James W. Hardin and Henrik Schmiediche*

Department of Statistics

Texas A&M University, College Station, Texas 77843

Abstract

Sophisticated data analysis requires both the use of a powerful data analysis language and the ability to display that data in a comprehensible yet visually stimulating manner. Many programs have been written to address either the data or graphical analysis needs of the researcher, but few satisfy both criteria. The *S* program by Becker, Chambers, and Wilks (1988) is a notable attempt to bridge the gap between these two different research needs. Yet *S* has limitations including very slow execution speed (in loops) and limited advanced graphics. An important point to be made at the outset is that a newer version of *S*, as yet unavailable to us, is said to have solved its memory leak and, thus, is much faster than the version to which we refer here. In this paper, we describe a new program that is our attempt to bring to researchers a fast and reliable method to perform data and graphical analysis on UNIX workstations. Its main features are: 1) A flexible and powerful interactive programming environment that is fast enough for simulation studies; 2) Extensibility via user defined functions; 3) Interactive graphics featuring two and three dimensional data plots, use of color as a fourth dimension, color image (surface) plots, 3-D color surface plots, rotating 3-D surface plots and color cycling; 4) sophisticated colorbar generation; and 5) publication quality graphics.

1. Introduction

One of the most popular data analysis tools on UNIX workstations for statisticians is the *S* program. This interactive language allows the user to easily calculate probabilities, obtain random numbers, write special purpose functions, and produce publication quality graphics. The expandability and permanent variables give the user the freedom to write powerful analytical tools, yet this power is sometimes used inefficiently. Loops are implemented as a matter of habit when much more efficient code may be used taking advantage of the fact that *S* supports matrix and array structures. Because of the poor execution time of *S* in loops and the fact that

the program grows unchecked in simulations, we decided to write an alternative program. An additional incentive was to provide a more complete graphical analysis package. In Table 1, we present evidence for the speed and memory efficiency of *MISHA* for programs that have loops. Experienced *S* users typically have invested time into building Fortran or C language object files that they link into *S* functions to handle these cases, but in *MISHA* the user can accomplish simulation type programs in a reasonable amount of time. In other comparisons, *S* and *MISHA* are similar in performance and *MISHA*, at this time, does not support dynamically linked object files though this support will be added. Note how the *S* program in Table 1 almost triples in size even though the only memory required by this function is for the *i*, *a*, and *b* variables. We believe that this is the reason behind why loops in this language execute so slowly. It was also the incentive for us to begin writing *MISHA*. Once the project started, we decided to include a number of different graphical analysis algorithms and to change the syntax to a C language format.

We contend that a good interactive program should exhibit the following five characteristics:

1. The user should have the ability to run programs in both batch and interactive mode.
2. The event of an error (syntactical, lexical, or computational) should result in a meaningful error message.
3. The user should be able to define functions and then be able to use them as easily as the program's builtin functions.
4. The program should allow data structures and numerical types that are intuitive and transparent.
5. The program should provide the means to have relational constructs, looping statements, and graphics.

2. Interactive Programs

The *S* program has enjoyed a good deal of popularity among statisticians and exhibits all of the properties mentioned in the last section with a small exception to

*James W. Hardin and Henrik Schmiediche are graduate students, Department of Statistics, Texas A&M University, College Station, TX 77843.

MISHA COMMAND		
<pre>for(i=1;i<=10000;++i) { a = cmat(rnorm(9),3,3) b = inv(a) }</pre>		
EXEC. TIME	START SIZE	END SIZE
14 seconds	252K	252K
S COMMAND		
<pre>for(i in 1:10000) { a <- matrix(rnorm(9),3,3) b <- solve(a) }</pre>		
EXEC. TIME	START SIZE	END SIZE
26.5 minutes	604K	1728K

Table 1: Comparison of execution time and program size for creating and inverting 10,000 random matrices in a loop as run on a Sun Sparc IPX. Sizes are taken from the `ps -u` command and reflect the size of the combined STACK and DATA segments. Comparisons are made with the January 1990 version of new S

property number 4. For example, if one attempts to calculate the square root of -1, an error message will appear on the screen even though the program supports complex data types. This is not to say that this is not a viable problem in S, however the user is first required to define a complex structure with -1 as the real part and 0 as the imaginary part and then a call to the square root function will return the appropriate solution. Our point is that since complex numbers are included in the program, one should not have to employ this extra step. TIMESLAB (Newton, 1988) has also enjoyed success among statisticians. The reasons for this are not limited to the fact that the program provides a valuable computational tool for those interested in time series analysis, but in many ways because users are able to interact with the program and to write macros. Additionally, this program, which differs from S in its origin, TIMESLAB was developed by a statistician while S was developed by a team of computer scientists at AT&T Bell Labs, is distinguishable from many statistician written programs in that the main impetus of the program does not limit the user to specifically that arena. The limitation with this program is that it is written for the IBM personal computer family and allows only 10,000 elements at any one time. Since some of the functions require workspace of twice the length of the data set, the user is actually limited to data of length much

less than 10,000 (NOTE: the author of TIMESLAB has ported much of his program in the form of S functions, where this limitation does not exist, and made them available on statlib). Another useful interactive program (first demonstrated at another Interface Symposium) is the MATRIX program (Alexander, 1989) which provided users with a powerful PC-based matrix oriented programming language. MATRIX supports user-defined functions and will output PostScript code. T_EX (Knuth 1986), although not a computational tool, is the embodiment of a good program. It provides an extremely valuable tool, contains thoughtful and entertaining documentation, and meaningful and sometimes humorous error messages—although some users may argue that after tracking down errors in a T_EX file, nothing is humorous.

3. Graphics

MISHA has a very powerful graphics engine and supports user defined plots. Graphics are drawn using internal objects and the user has access to any and all of these objects in order to define a layout for a plot. MISHA graphics can be displayed interactively under X (X Window System, Version 11 Release 4) and in color PostScript to produce a hard copy of the plot. Under X, MISHA will allow the user to display an arbitrary number of plots simultaneously (limited only by computing resources). MISHA makes every attempt to produce a WYSIWYG (what-you-see-is-what-you-get) plot even to the point of scaling line thickness and fonts to the appropriate sizes when the plot window is interactively resized. Some compromises had to be made since X does not support continuous font scaling (MISHA will use the closest available font size) and font rotation.

3.1. Graphics Engine

The heart of MISHA graphics is a graphics engine that allows for the creation of sophisticated plots using an object oriented approach. The graphics engine manipulates two basic kinds of objects which we will refer to as *graphics objects* and *graphics drawables*.

A graphics object is simply a low level graphics command to perform some graphics related action. For example, a graphics object might draw a line between two points or it might place text on the plot. Any graphics object can have several attributes that define the precise composition of the object. For example, a simple line between two points will have the following attributes: 1) (x_1, y_1) and (x_2, y_2) , 2) line width, 3) line color, 4) line style (solid, dashes, etc.) and so on. Many of these individual graphics objects are chained together to form the desired plot.

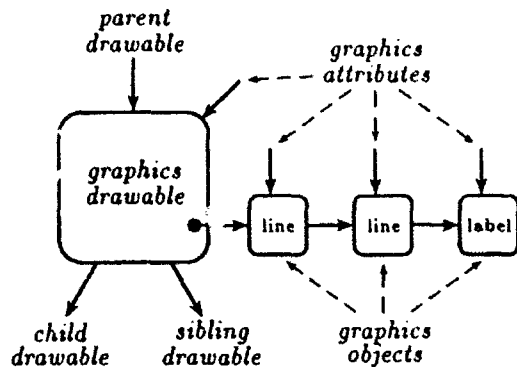


Figure 1: A MISHA graphics drawable with a chain of three graphics objects. The drawable defines the screen parameters, while the graphics objects draw the desired picture.

The second basic type of graphics object is the *graphics drawable* which, in a sense, provides the canvas on which MISHA "paints" or draws the plot. More specifically it serves the following purposes: 1) it is the beginning or "anchor" to a chain of graphics objects; 2) it defines an area on the screen on which these graphics objects will function; 3) it defines the coordinate system for its graphics objects and 4) it performs any necessary graphics clipping. In addition to these basic functions, the graphics drawable also has a set of attributes like foreground and background color, transparency or opacity, etc. Figure 1 is a visual representation of a drawable and its corresponding graphics objects. Any drawable can contain any number of multiple descendant or *child* drawables. Each descendant or *child* drawable with the same parent drawable are considered *siblings* of each other. Figure 2 shows what a directed tree of drawable graphics objects might look like. For more information on *directed trees* see Lewis and Smith, 1982 or Knuth, 1973.

Using these two basic types of graphics objects, the MISHA graphics engine simply constructs any desired plot through the following series of steps:

1. Initialize the current drawable (beginning with the root drawable).
2. Process the graphics objects of the current drawable.
3. Do the first possible of the following choices:
 - (a) Go to the next child drawable and repeat all steps.
 - (b) Go to the next sibling drawable and repeat all steps.

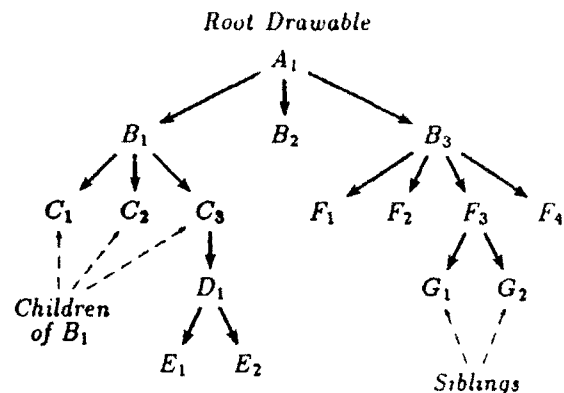


Figure 2: A MISHA directed tree of graphics drawables. Each drawable could contain a chain of graphics objects. The MISHA graphics engine would traverse the tree in the following order: $A_1, B_1, C_1, C_2, C_3, D_1, E_1, E_2, B_2, B_3, F_1, F_2, F_3, G_1, G_2, F_4$.

- (c) Go to the parent drawable and repeat step 3.
- (d) Exit.

This process insures the entire tree is traversed and all graphics objects are drawn.

Even though only one drawable is necessary to construct any given plot, the ability to link graphics drawables together in a tree like fashion gives MISHA graphics increased flexibility and power. It allows the user and/or programmer of MISHA graphics to break a graphic or plot into reusable component parts. For example, a standard two way scatter plot generally consists of at least three parts—the data points, the x-axis and the y-axis. MISHA creates three separate drawables, with arbitrary internal coordinate systems, that create the x-axis, y-axis and the data driven scatter plot. The final plot would consist of a graphics drawable that places these three drawables in their respective positions within the parent drawable to create the final plot. The advantage here is that many plots make use of the above three graphics components and, once programmed, they are easily reusable. For example, an $n \times m$ matrix plot could be constructed by chaining the scatter plot drawable $n \times m$ times off of a root or parent drawable. We can visualize this process as pruning and grafting branches of MISHA graphics trees in new ways to construct new plots. An additional advantage to this approach to graphics is that any changes that are made to individual drawables will automatically be available to all graphics that uses them. Note that this is also true for individual graphics objects which is a basic feature of object oriented design.

MISHA's method of displaying graphics might seem

Misha 2D Plot

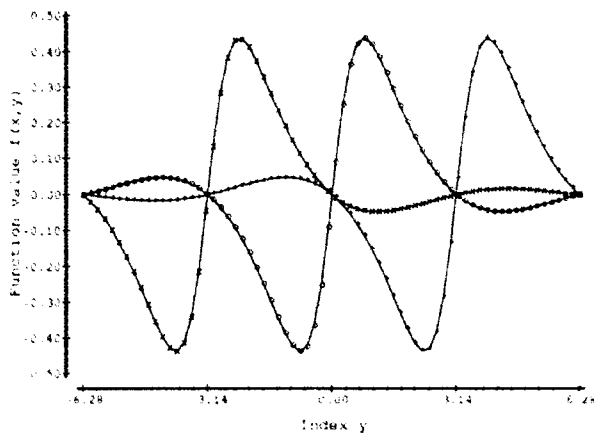


Figure 3: A MISHA plot. A plot of the function described in figure 4 where $x: -2\pi \leq x \leq 2\pi$ and $y = -\pi, 0, \pi$.

daunting and complicated at first. Note, however, that actually displaying a plot that has already been programmed into MISHA is easy and generally requires only one MISHA command. The power and flexibility of the graphics engine comes into play when new plots need to be designed while classic plot layouts are included as commands in the language.

3.2. Graphics Objects

MISHA provides a variety of basic graphics objects to construct plots and other graphics. Some MISHA graphics objects perform tasks like drawing a line between two points, drawing a sequence of connected lines, placing some text anywhere on the screen using numerous fonts (any font available in postscript), drawing rectangles and so on. Using these basic objects MISHA with a single command, can create a standard line plot like the one presented in figure 3.

Some of the more advanced MISHA graphics are generated using specialized image and 3-D surface graphics objects. The image graphics object displays an image plot of three dimensional matrix data. It maps the two index data dimensions onto the x-y plane and maps the third dimension onto a color scale. An image plot is perhaps best described as a *color contour* plot. Figure 4 is such an image plot where the color scale is from grey to black. A grey to black color scale does not do justice to the visual impact that such a plot can produce. The image graphics objects will optionally smooth the data to generate smooth contour lines. If it did not, image plots for low resolution data would have a block like appearance and contours would be jaggy and difficult to follow. The smoothing algorithm MISHA uses is based on univariate cubic spline interpolation and was developed by Graham Dunnett (1992, personal commu-

Misha Surface Plot

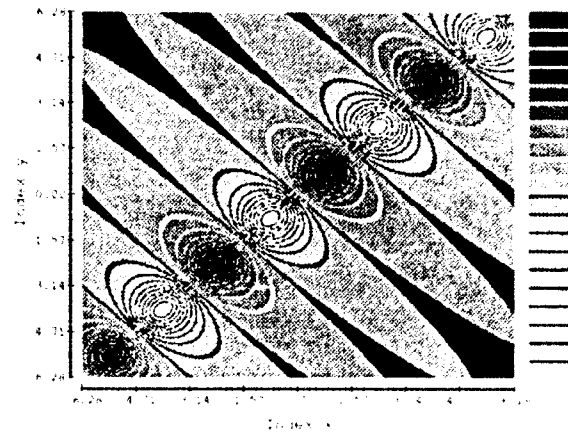


Figure 4: A MISHA image plot. The function plotted is $f(x, y) = \sin(x + y)/(1 + (x - y)^2)$ where x and y go from -2π to 2π .

nication) and is based on work by Yngvesson and Wallin (1991) and Jones and Saupe (1991). The main reason for the choice of this algorithm is speed. The algorithm, coded in C, can interpolate 120,000 data points (pixels) in under a second on a workstation like the SPARC IPX with excellent results. In an interactive MISHA session the user could additionally cycle the color scale. If the color scale is set up properly, this amounts to animating contour lines over small changes allowing for a visually stunning and very informative insight into the precise contour structure of the data.

The 3-D surface graphics object three dimensionally projects an image plot on the screen. This is similar in nature to the 3-D mesh plots that some programs generate. Figure 5 is an example of such a 3-D surface plot. In an interactive MISHA session, the user could rotate the 3-D surface plot in real time. An additional feature of these 3-D surface plots is the ability to map a fourth dimension of data to the color scale with height, depth and width being the first three data dimensions. Color cycling is also possible.

3.3. Color

Color can be used in MISHA to enhance the visual appeal of a plot. Any and all graphic objects that draw something will allow the user to specify a color. By default, graphics objects will use the foreground and background colors of the graphics drawable to which they are chained. Thus, changes in foreground and background colors will propagate to the individual graphics objects. Color is specified in MISHA by name, from a 700 color name database, or using any one of three common color models: RGB (Red-Green-Blue), HLS (Hue-Lightness-Saturation) and HSV (Hue-Saturation-Value).

Misha 3D Surface Plot

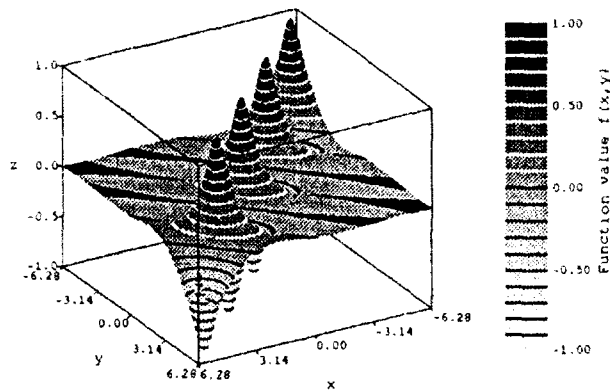


Figure 5: A MISHA 3-D surface plot. Same function as figure 4 where $z = f(x, y)$.

The RGB color model is used by most CRT display technologies, while the HLS and HSV color models are *perceptual* color models that are designed to be more intuitive to most users. For more detailed information on the nature of these color models see Foley and Van Dam (1984).

In image and 3D surface plots, color does more than just enhance the visual appeal of the plot. Here color represents part of the data and information that is presented, thus the choice of color scale or colorbar is vital to the overall design of the plot. To help users create and select colorbars, MISHA maintains a colorbar database that is easily modified and expanded by the user to meet different criteria and needs. Colorbar creation is straight forward and encourages experimentation. Even though a user could specify every individual color in a colorbar, MISHA allows a user to specify one or more ranges of colors over which to interpolate to create a color scale. The interpolation color space can be HLS, HSV or RGB and is independent of the color space used to specify the colors.

4. Conclusion

Those users familiar with writing TIMESLAB macros, GAUSS functions, or S functions will be able to write functions in the MISHA language. Since MISHA uses standard function names, users should be able to port functions between these 4 packages with a minimum number of changes. MISHA will also offer high level commands and functions that will ease the burden of the user to obtain complete standard analyses. Among the high level commands to be implemented include standard ANOVA and regression functions that will return

an array of diagnostics along with the usual estimates and associated *p*-values.

As the program matures, more of these types of functions will be added. One of the most important features to be offered is the ability of the user to interact with graphics. This program will offer a new tool for the computational statistician and will continue to grow and expand directed by its users. With the speed and graphical abilities in place, time will be taken to expand the intrinsic functions to reflect a more diverse arena of scientific thought.

5. References

- Alexander, W.P. (1989), "MATRIX: An Interpreted Matrix Language", *Computing Science and Statistics: Proceedings of the 21st Symposium on the Interface*, ASA.
- Becker, R.A., Chambers, J.M., and Wilks, A.R. (1988), *The New S Language*, Pacific Grove, CA, Wadsworth & Brooks/Cole.
- Foley, J.D. and Van Dam, A. (1984), *Fundamentals of Interactive Computer Graphics*, Reading, Massachusetts, Addison-Wesley Publishing Company.
- Jones, Huw and Saupe, Dietmar. (1991), "Stochastic Methods and Natural Phenomena", *Eurographics*, 2.
- Knuth, Donald E. (1973), *The Art of Computer Programming: Volume 3 / Sorting and Searching*, Reading, Massachusetts, Addison-Wesley Publishing Company.
- Lewis, T.G. and Smith, M.Z. (1982), *Applying Data Structures* Second Edition, Boston, Massachusetts, Houghton Mifflin Company.
- Newton, H.J. (1988), *TIMESLAB: A Time Series Analysis Laboratory*, Pacific Grove, CA, Wadsworth & Brooks/Cole.
- Yngvesson, Jonas and Wallin, Inge (1991), *Public domain rendering software*. Linkoping Institute of Technology, Sweden.

Interactive Analysis of Gappy Bivariate Time Series Using AGSS

Peter A. W. Lewis and Bonnie K. Ray

Dept. of Operations Research, Naval Postgraduate School
Monterey, CA 93943

Abstract

Bivariate time series which display nonstationary behavior, such as cycles or long term trends, are common in fields such as oceanography and meteorology. These are usually very large scale data sets and often may contain long gaps of missing values in one or both series, with the gaps perhaps occurring at different time periods in the two series. We present a simplified but effective method of interactively examining and filling in the missing values in such series using extensions of the methods available in AGSS, an APL2-based statistical software package. Our method allows for possible detrending and removal of seasonal components before automatically estimating arbitrary patterns of missing values for each series. Interactive bivariate spectral analysis can then be performed on the detrended and deseasonalized interpolated data if desired. We illustrate our results using a bivariate time series of ocean current velocities measured off the California coast.

1 Introduction

Gaps of missing values of various sizes are a common problem in many data sets. In oceanographic data, for example, a single large gap may arise in the gathering of tidal data when an instrument stops working and the malfunction is not detected for several days. Many small gaps are more characteristic of data gathered from satellites. The missing value problem is complicated for bivariate series in that the gaps may not fall at the same time periods in both series. Ad hoc univariate methods, such as basing "suitable" replacement values on the range of values assumed by neighboring points or points of the same periodicity, fail to account for possible cross correlation in the data. In order to successfully analyze the spectrum of gappy data sets, or use the data for other purposes, the missing values need to be estimated in a way that is characteristic of the rest of the bivariate data set.

The problem of missing values in time series has been studied by several authors in recent years, primarily in a state space framework. Jones (1980) used a Kalman filter recursion to calculate the exact likelihood of a univariate stationary autoregressive moving average (ARMA) process with missing values, while Harvey and Pierce (1984) and Kohn and Ansley (1986) extended the Kalman filtering method to nonstationary autoregressive integrated moving average (ARIMA) processes. Ansley and Kohn (1985) gave a method of recursively calculating the likelihood for a multivariate state space model with incompletely specified initial conditions which can be used to interpolate an arbitrary pattern of missing values in multivariate time series. Both the computation and derivation are much simpler in the univariate case than in the multivariate case. More recently, Ljung (1989) derived an exact expression for the estimates of missing values in a univariate ARIMA process in a form that is useful for examining the estimates and their mean squared errors. For an arbitrary pattern of missing values, however, the computations are not very efficient. None of the above methods is thus easy to implement in practice for bivariate series which are possibly nonstationary and have arbitrary patterns of missing values in both series.

In this paper, we present an algorithm for semi-automatically filling in gaps in bivariate time series, allowing for trends, cycles, and cross correlation in the data. The interactive implementation of the algorithm allows for visual examination of the data at each step, giving the practitioner the opportunity to view the original data with missing values, the "patched" data in which the missing values have been filled in using linear interpolation, and the estimated autospectrum of the crudely interpolated data. After this examination, one may choose to remove a trend or cycles from the data. The remaining series is automatically modeled as an autoregressive process and the estimated model is used for interpolation. The method allows

for joint interpolation of two correlated series, incorporating an estimate of the autocorrelation for each series, and the cross correlation between the two series, into the interpolated values. At the end of the interpolation phase, the user has the choice of examining the coherence function of the interpolated series, as well as producing more detailed plots of particular segments of the series. The following section presents the interpolation algorithm in detail, while Section 3 gives an application of the algorithm to a bivariate series of ocean current velocity meter readings measured off the California coast.

2 Algorithm

The following algorithm has been coded in APL2 using the IBM APL2 AGSS program as a computing platform. Thus functions such as regression, the Fast Fourier Transform used for computing the periodogram of the series, and random number generation from AGSS are used, as well as some of the AGSS graphics screens. The algorithm is available in a AGSS library from the authors for mainframe or microcomputer data. The algorithm is as follows:

1. The user is asked to enter the name of the original series containing gaps and the series is plotted. Denote this series by $x(t)$, $t = 1, 2, \dots, n$. If there are two series containing gaps, and the two series are cross correlated in some way, the user is asked to enter the name of the second series and the second series is plotted as well. Denote this series by $y(t)$, $t = 1, 2, \dots, n$. The two series must have the same length, but the location and length of gaps in the series may differ.
2. The user is asked to enter the number used on the data record to indicate a missing value.
3. The program then computes the locations of the gaps and fills in the missing values for each series by linearly interpolating between the two points on either side of the gap. Denote the linearly interpolated series as $x_1(t)$ and $y_1(t)$ respectively. The resulting series are then plotted and can be visually examined by the user to decide whether removal of a linear trend is necessary.

Note: Steps 4-8 are applied to both the $x_1(t)$ and $y_1(t)$ series in exactly the same manner. Only the results for the $x_1(t)$ series are given below.

4. If so desired, the program removes a linear trend from each series. The trend is estimated using

least squares applied to the initial "patched" series. The resulting series is

$$x_2(t) = x_1(t) - \hat{a} - \hat{b}t,$$

where \hat{a} is the estimated constant and \hat{b} is the estimated slope.

5. The sample periodogram is automatically estimated and plotted for the interpolated and detrended data, $x_2(t)$. (see, for example, Priestly (1981) for a definition of the periodogram and its interpretation).
6. The program calculates the probability of obtaining the computed values for the 20 largest values of the normalized periodogram under the assumption that $x_2(t)$ is a Gaussian white noise process. A small probability indicates that a cycle may be present in the data. The probabilities are computed using the large sample test statistic for I_j , $j = 1, 2, [n/2]$, where I_j denotes the j^{th} largest ordinate of the periodogram. (Priestly 1981, p.407).
7. Using the information obtained in the previous step, as well as any intuitive or physical knowledge of cyclic behavior in the series, the user specifies cycles to be estimated and removed from the interpolated and detrended data, if desired. The cycles are assumed to be of the form

$$s_t = \sum_{j=1}^J \{ \gamma_j \cos(\omega_j t) + \beta_j \sin(\omega_j t) \},$$

where $\omega_j = 2\pi f_j$ are the frequencies that you would like to remove and J is the number of cycles. The γ_j and β_j are estimated using least squares. The resulting series is $x_3(t) = x_2(t) - \hat{s}_t$.

8. A first order autoregressive (AR) model is fitted to the detrended and deseasonalized data $x_3(t)$. An AR(1) model has the form

$$x_3(t) = \phi x_3(t-1) + a_x(t), \quad t = 2, 3, \dots, n.$$

We assume that $a_x(t) \sim iid N(0, \sigma_{a_x}^2)$. The parameter ϕ is estimated using least squares. The residual series $\hat{a}_x(t)$ is

$$\hat{a}_x(t) = x_3(t) - \hat{\phi} x_3(t-1), \quad t = 2, 3, \dots, n.$$

9. The variance of $\{\hat{a}_x(t)\}$ is calculated and a white noise series of length n having the distribution $N(0, \hat{\sigma}_{a_x}^2)$ is generated. Denote this series by $a'_x(t)$.

10. Let l denote the length of a particular gap in the series and let $x_3(t)$ and $x_3(t+l+1)$ denote the points on either side of the gap. The program forecasts and backcasts from each end of the gap using the following recursive equations for $j = 1, 2, \dots, l$.

$$\begin{aligned}\hat{x}_3(t+j) &= \hat{\phi}\hat{x}_3(t+j-1) + a'_x(t+j) \\ \hat{x}_3(t+l+1-j) &= \hat{\phi}\hat{x}_3(t+l+2-j) \\ &\quad + a'_x(t+l+1-j).\end{aligned}$$

Then the interpolated value is

$$\tilde{x}_3(t+j) = w_{1j}\hat{x}_3(t+j) + w_{2j}\hat{x}_3(t+l+1-j),$$

where $w_{1j} = 1 - (j/l+1)$ and $w_{2j} = 1 - w_{1j}$.

11. If interpolating values for two correlated series, the standard deviation of the residual series $\{\hat{a}_x(t)\}$ and $\{\hat{a}_y(t)\}$ found in Step 8 is calculated and the sample cross correlation at lag 0 between $\{\hat{a}_x(t)\}$ and $\{\hat{a}_y(t)\}$ is computed. Denote this by c . A white noise series of length n having the distribution $N(0, \hat{\sigma}_{a,y}^2)$ is generated. Denote this series by $a'_y(t)$. A second white noise series of length n is generated using the following relation:

$$a''_y(t) = c(\hat{\sigma}_{a,y}/\hat{\sigma}_{a,x})a'_x(t) + \sqrt{1-c^2}a'_y(t).$$

12. The values for the $y_3(t)$ series are interpolated as in Step 9, with $a'_x(t+j)$ replaced by $a''_y(t+j)$
13. The estimated trend and cycle are added back to the interpolated series and the series containing the final estimates of the missing values is plotted.
14. The user may choose to plot the coherence function for the detrended and deseasonalized interpolated series if desired.
15. The user may also choose to plot more detailed segments of the final interpolated series if desired.

Using a weighted average of backward and forward forecasts made from each end of a gap using an estimated univariate ARMA model, as was used in Step 10, was discussed by Abraham (1981). He calculates the weights to minimize the mean-squared error of the interpolated value, thus the weights depend in a complicated way on both the estimated model parameters and the length of gap. Our method, although simple, is intuitively appealing in that it gives less weight to interpolated values at long lead times and is easy to implement when the lengths of the gaps are different. Note that in Step 10, we include a simulated

noise term in the usual expressions for the backward and forward forecasts of an AR(1) model. This is to eliminate unrealistic "smoothness" in the interpolated values, which will occur if the gap is very long. Realistic noise in the series is important if the interpolated series will be used to estimate the spectral density of the series. Similarly, in Step 11 we incorporate an estimate of the contemporaneous correlation between the two series into the estimates of the interpolated values of the second series by including a noise term taken from a simulated bivariate Normal pair of series with correlation c . The method for generating a bivariate Normal pair with specified correlation is taken from Lewis and Orav (1989, p.301). A discussion of contemporaneous bivariate time series models made be found in Camacho, Hipel, and McLeod (1987).

3 An Example

We apply the above algorithm to a vector pair of ocean current velocities collected off Point Sur, California over the period 0000 hours, Sept. 19, 1990 to 2300 hours, Oct. 30, 1990, a total period of 1008 hours. Current velocities are just one set of variables which are collected regularly by oceanographers at the Naval Postgraduate School in order to provide information related to the long term variability in sea surface temperatures off the California coast. The velocities were measured using a paddlewheel and electronic counter assembly located at the top of the recording unit placed at a depth of 350 meters. Velocity, in units of cm/s, was determined from the number of revolutions made by the paddlewheel during each sampling interval, to an accuracy of ± 1.0 cm/s. The data was initially recorded at 30 minute intervals. After initial visual inspection for outliers or periods suspect of instrument failures, and manual editing if necessary, the data was filtered using a Cosine-Lanczos filter with a centered 25 point data window and interpolated to specified 60 minute intervals. At this point, there remained a period of 63 consecutive hours of missing data, in which the data gathering instruments were not working. There were also a few scattered individual missing values.

Figures 1 and 2 show plots of the E-W (or u) component of the current velocity, and the N-S (or v) component of the current, with missing values coded as 0. Figures 3 and 4 depict the same series with missing values "patched" using linear interpolation. There appear to be regular cycles in the data, as expected for current data, as well as the presence of a long term trend. We fit a linear trend to both the u and v cur-

rent components, with estimated constants of 3.91 and 1.60, respectively, and estimated slope coefficients of -0.0018 and -0.0003. Standard t -tests on the significance of the regression coefficients are not appropriate because the residuals are not assumed to be uncorrelated. Figures 5 and 6 show the periodogram for each detrended series. Diurnal and semi-diurnal cycles are clearly indicated for the v -component of current velocity, with only the diurnal cycle clearly seen in the u -component. In addition, there appears to be some long range dependence in the data, as evidenced by the large values of the periodogram at small frequencies, which remain even after the removal of a long term trend. See Lewis and Ray (1992) for a discussion of long range dependence in sea surface data. Based on the approximate p -values of the test statistic for each of the 20 largest periodogram ordinates and knowledge of the tidal cycles, we remove cycles at frequencies (in cycles per 1008 hours) 81, 82, and 84 in the u -component and frequencies 42, 81, 82, and 84 in the v -component. Table 1 gives the values of the normalized periodogram values at these ordinates and the resulting p -values for the test statistic. After this step, the missing values are automatically estimated for the detrended and deseasonalized data. Figures 7 and 8 show the two series with final estimates of the missing values, after the trend and cycles have been added back to the series. The estimated values appear to follow the pattern of the data quite nicely.

We study the correlation between the two series in more detail by looking at the coherence function for the two interpolated, detrended and deseasonalized series. The coherence is initially computed assuming a cosine arch window of length 7. The user has the option of changing the window at any point in the coherence analysis. Figure 9 shows the cross-phase, cross-amplitude, and cross-coherence functions of the two series. The cross-amplitude spectrum shows dependence between the series at fairly low frequencies, but the (normalized) coherence measure shows that the dependence extends to high frequencies as well. No systematic effect can be seen in the phase spectrum.

Additionally, the enlarged segment in Figure 10 depicts the two series with values plotted as vertical lines drawn from the x -axis. The segment partially includes the interpolated values shown in Figures 7 and 8. No apparent difference between the known and the interpolated values can be seen.

4 Summary

We have presented a simple algorithm which permits interpolation of arbitrary patterns of missing values in both univariate and bivariate time series, allowing for the possibility of non-stationarity. The implementation is interactive and has graphical capabilities available at each step. It is also much easier to implement in practice than the state-space approach of Ansley and Kohn(1985), which requires modified versions of the Kalman filter and the fixed point smoothing algorithm. The estimated contemporaneous correlation between the two series is used in the interpolation algorithm in order to estimate the missing values in a manner that is consistent with the rest of the data. Although we have assumed that each series follows a simple AR(1) model, the algorithm could easily be extended to model each series as an ARMA(p, q) model, with the orders of p and q chosen by the user after examination of the sample autocorrelation and partial autocorrelation functions of the detrended and deseasonalized "patched" data. Functions necessary to compute the sample correlation functions and estimate the parameters of an ARMA(p, q) model are already present in the IBM APL2 AGSS package. (The AGSS application discussed here is available from the authors; the AGSS package is available from IBM.)

References

- Abraham, B. (1981), "Missing observations in time series", *Commun. Statist.-Theor. Meth.*, A10(16), 1643-1653.
- Ansley, C. F. and Kohn, R. (1985), "Estimation, filtering, and smoothing in state space models with incompletely specified initial conditions," *Annals of Statistics*, 13, 1286-1316.
- Camacho, F., McLeod, A. I., and Hipel, K. W. (1987) "Contemporaneous bivariate time series," *Biometrika*, 74(1), 103-113.
- Harvey, A. C., and Pierse, R. G. (1984), "Estimating missing observations in economic time series," *Journal of the American Statistical Association*, 79, 125-131.
- Jones, R. H. (1980), "Maximum likelihood fitting of ARMA models to time series with missing observations," *Technometrics*, 22, 389-395.

Kohn, R. and Ansley, C. F. (1986), "Estimation, prediction, and interpolation for ARIMA models with missing data," *Journal of the American Statistical Association*, 81(395), 351-361.

Lewis, P. A. W. and Orav, E. J. (1989) *Simulation Methodology for Statisticians, operations Analysts, and Engineers*, Pacific Grove: Wadsworth & Brooks/Cole.

Lewis, P. A. W. and Ray, B. K. (1992), "Modeling non-linear time series with long range dependence," Technical report, Naval Postgraduate School.

Ljung, Greta M. (1989), "A note on estimation of missing values in time series," *Commun. Statist. - Simula.*, 18(2), 459-465.

Priestley, M. B. (1981) *Spectral Analysis and Time Series*, London: Academic Press.

Tables and Figures

Table 1: Normalized Periodogram Values for Ocean Current Velocities

U-component of current velocity		
Frequency	Norm. Periodogram Value	p-value
1	106.14	0.00
81	88.52	0.00
2	35.67	0.00
82	22.42	0.00
84	20.85	0.00
V-component of current velocity		
Frequency	Norm. Periodogram Value	p-value
81	195.51	0.00
1	69.88	0.00
84	34.81	0.00
2	20.90	0.00
82	14.40	0.00
42	10.09	0.11

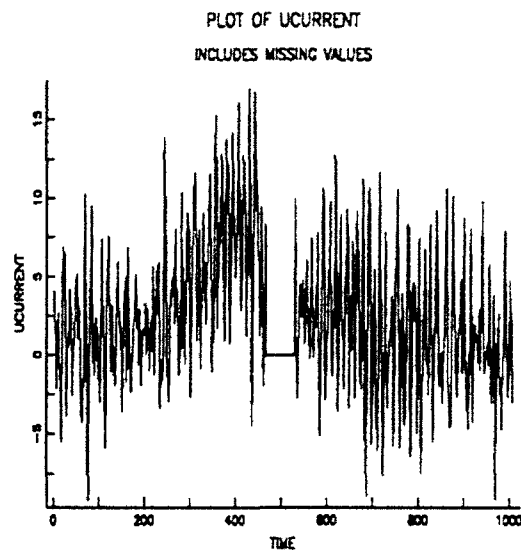


Figure 1: U-component of Current Velocity (missing values coded as 0's)

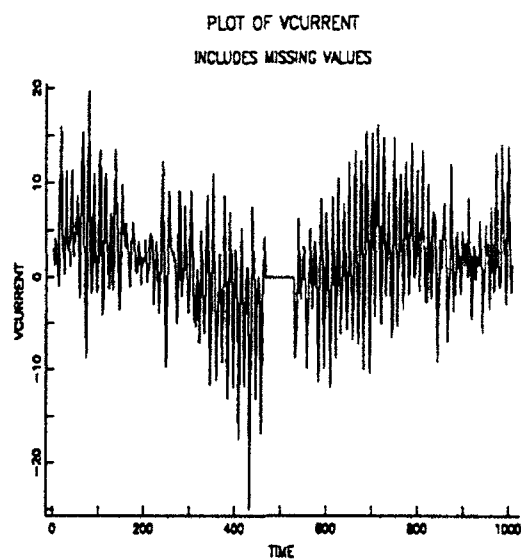


Figure 2: V-component of Current Velocity (missing values coded as 0's)

PLOT OF UCURRENT WITH INITIAL ESTIMATES OF MISSING VALUES

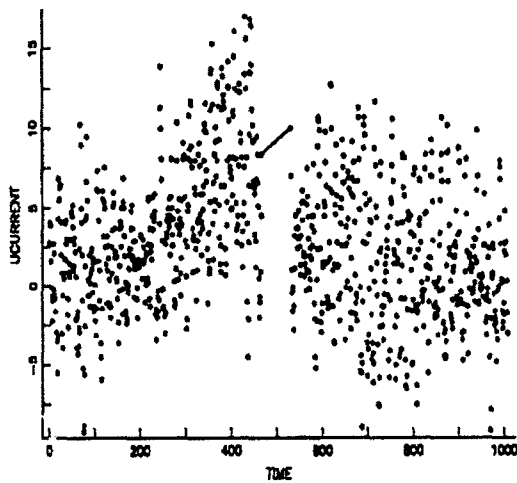


Figure 3: U-component of Current Velocity (missing values linearly interpolated)

ESTIMATED SPECTRAL DENSITY FOR UCURRENT

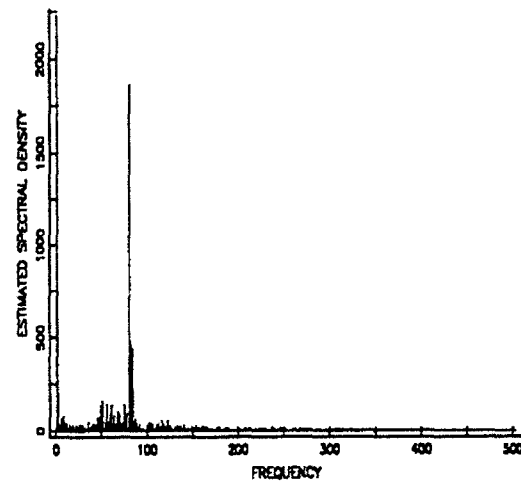


Figure 5: Periodogram of U-component of Current Velocity after linear detrending

PLOT OF VCURRENT WITH INITIAL ESTIMATES OF MISSING VALUES

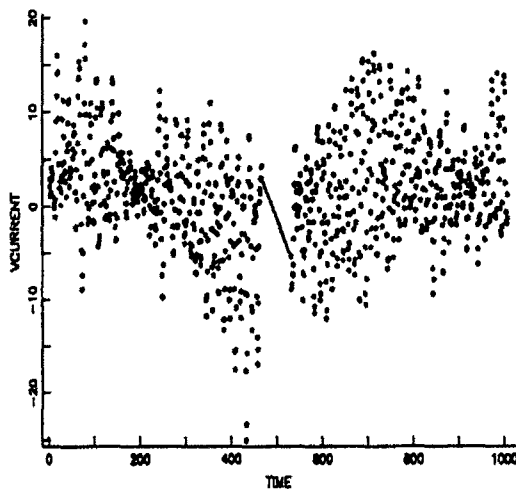


Figure 4: V-component of Current Velocity (missing values linearly interpolated)

ESTIMATED SPECTRAL DENSITY FOR VCURRENT

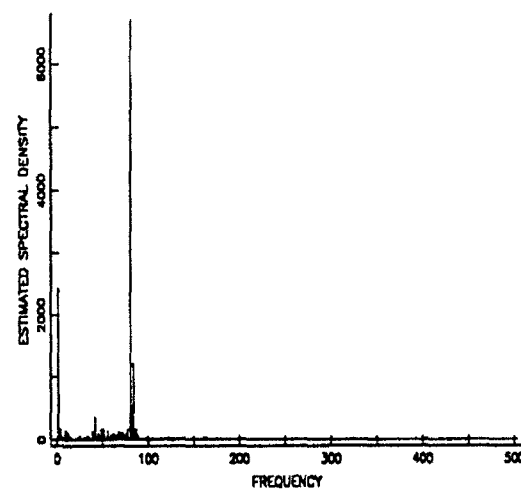


Figure 6: Periodogram of V-component of Current Velocity after linear detrending

PLOT OF UCURRENT WITH FINAL ESTIMATES OF MISSING VALUES

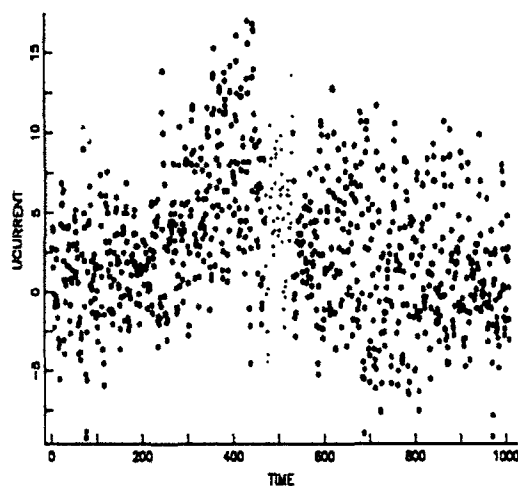


Figure 7: U-component of Current Velocity (missing values estimated using complete interpolation procedure)

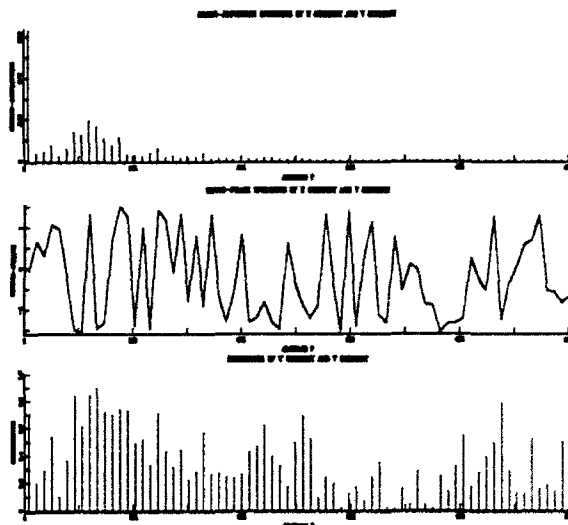


Figure 9: Cross-amplitude Spectrum (top), Phase Spectrum (middle), and Coherence (bottom) of U-component and V-component of Current Velocity

PLOT OF VCURRENT WITH FINAL ESTIMATES OF MISSING VALUES

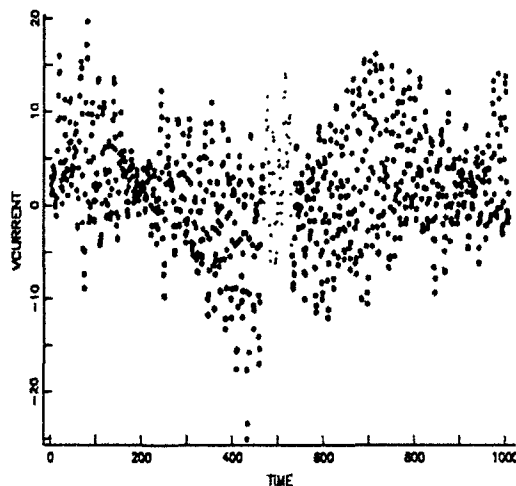


Figure 8: V-component of Current Velocity (missing values estimated using the complete interpolation procedure)

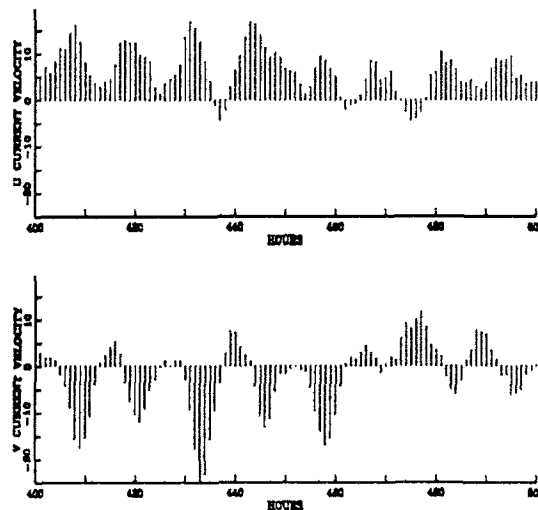


Figure 10: Detailed segment of U-component and V-component of Current Velocity after final estimation of missing values

A COMPARATIVE VISUALIZATION STUDY OF A GRAPHICAL MULTIVARIATE NORMALITY GOF TEST.

Jorge Luis Romeu
Department of Mathematics
SUNY Cortland, NY 13045
jromeu@svm.bitnet

Abstract

An ordered sequence of plots of a new graphical multivariate normality test, performed on samples of selected distributions, is presented. The multivariate sample is transformed into a set of linked vectors in a bivariate space. According to where the vector endpoints fall, in relation to the confidence ellipse, multivariate normality is accepted or rejected. If normality is rejected, we visually analyze where the vector's endpoints fall, outside the confidence ellipse. We also compare the linked vector pattern, in relation to the null (solid line) pattern. This visual analysis provides an indication of how does the alternative non normal distribution looks like.

1.0 The Graphical Test.

The Multivariate Q_n (Ozturk and Romeu, 1992) graphical procedure can be divided into three parts: (i) the confidence ellipse, (ii) the lower half of the pattern and (iii) the upper half (Figure 1). When a sample comes from a multivariate normal distribution, the corresponding linked vector closely follows both halves of the pattern and ends within the confidence ellipse. When the sample comes from another distribution, its vector endpoint falls outside the confidence ellipse. But the area, outside of the ellipse it falls, and the pattern it follows, is closely associated to the distribution it comes from. In this paper we present a study of test patterns from non normal alternative distributions.

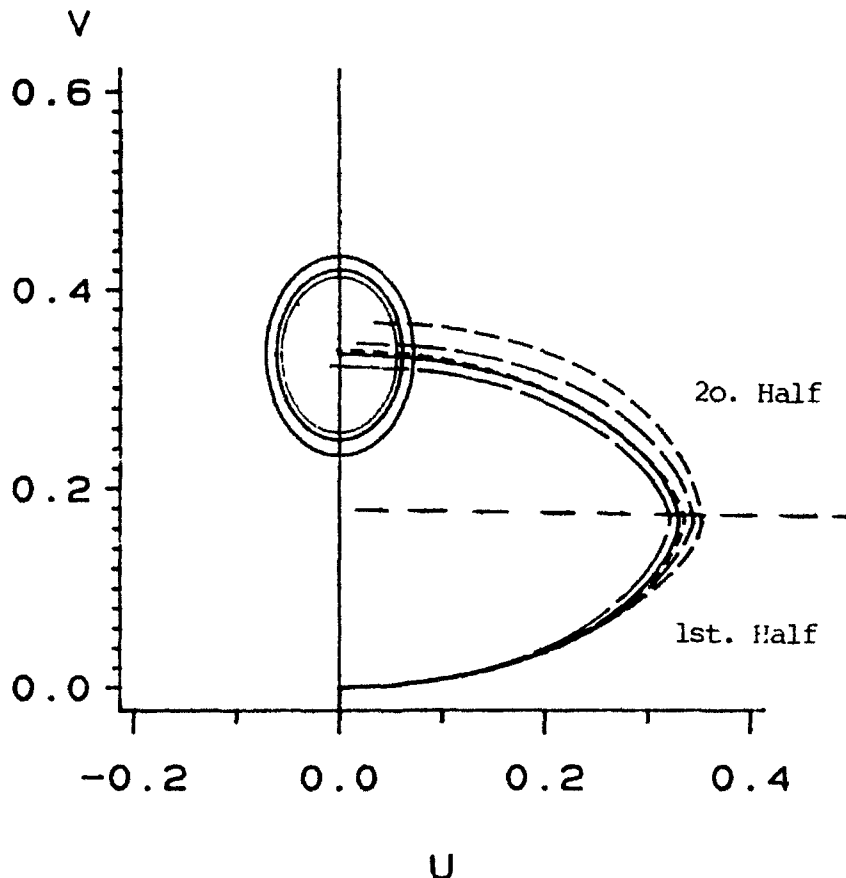


Figure 1.

2.0 Non Normal Alternatives.

The bivariate distributions in this study (Figure 2) were chosen as to be (i) more skewed, (ii) more kurtic than a bivariate normal and (iii) a combination of these. In this paper we only present results for three special cases: skewed, kurtic and combination. Samples of size $n=100$ were drawn and submitted to our graphical test.

We used the Generalized Lambda Distribution (GLD) to obtain an increasing sequence of bivariate skewed distributions. We chose the bivariate Uniform to have a flatter distribution than the bivariate normal. We chose the bivariate T with 8 d.f., to have a peaked distribution. Both of these were purely Kurtic (i.e. no symmetry problems).

Finally, we chose the Chi Square distribution with 10 d.f. as one which would be skewed and kurtic at the same time. But the degree of skewness would be consistent with that of the GLD used. And the degree of kurtosis, with that of the t distribution.

We sampled these distributions extensively in a Monte Carlo power study for our test (Romeu, 1990). In this paper, we undertake a visual study of the patterns obtained when samples come from such distributions.

Existing multivariate normality GOF tests are not graphical. Ours allows the user not only to accept or reject, but to obtain a sense of where to go next (i.e. what does the alternative distribution look like) in the last case.

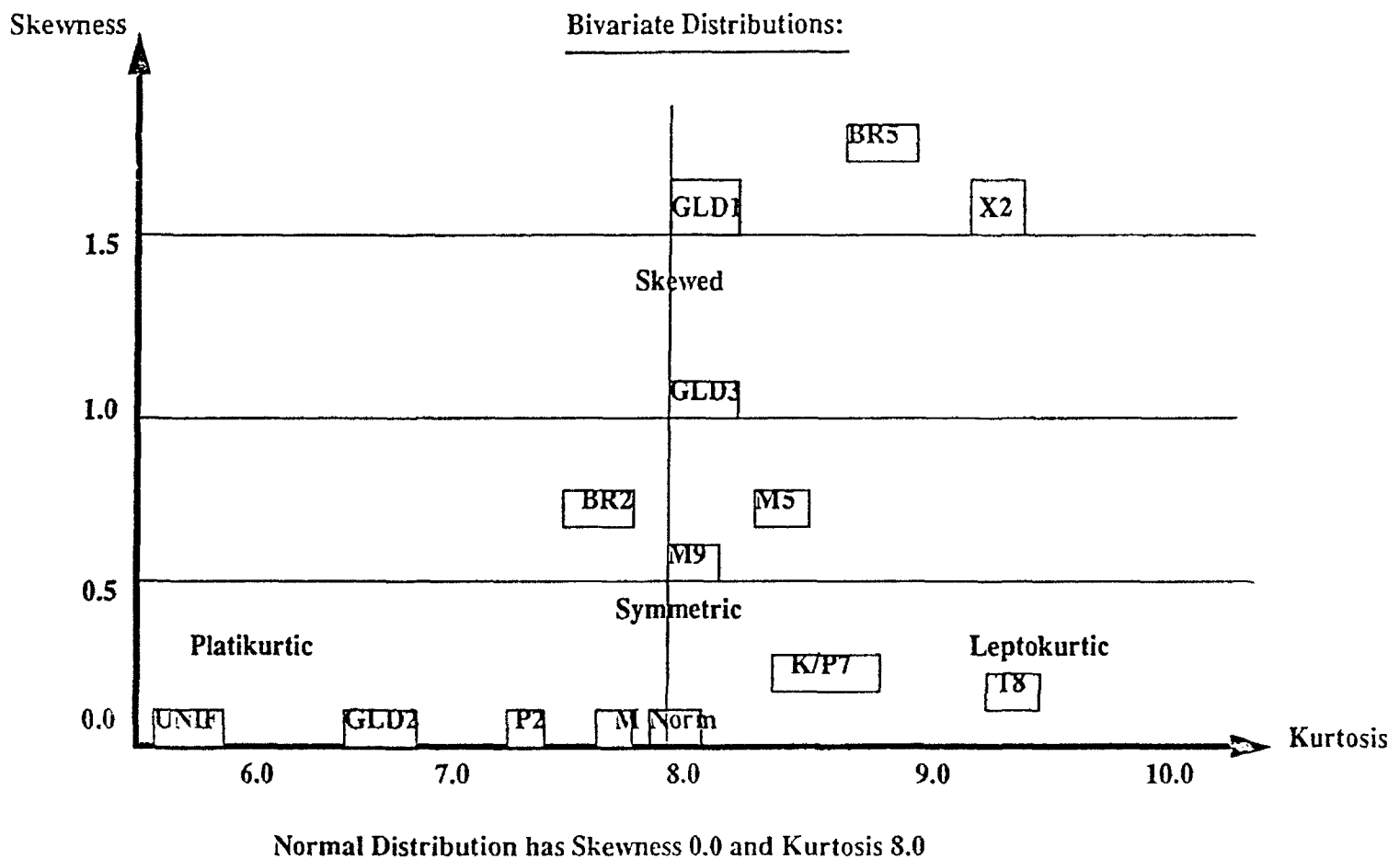


Figure 2. Statistical Distributions in the Skewness vs. Kurtosis plane.

3.0 Purely Skewed Distributions.

The pattern shown in Figure 3 corresponds to a sample taken from GDL1, a bivariate distribution with skewness of 1.5 and kurtosis of 8.0 (purely skewed). We see how the endpoint of the sample linked vector falls off the upper left quadrant of the confidence interval. This allows us to reject bivariate normality at all (90%, 95%, 99%) levels.

In addition, we observe the distinct pattern of the sample linked vector, sharply increasing in the first half, then changing direction before ever crossing the null distribution pattern.

We can use this test pattern to recognize a purely skewed non normal alternative.

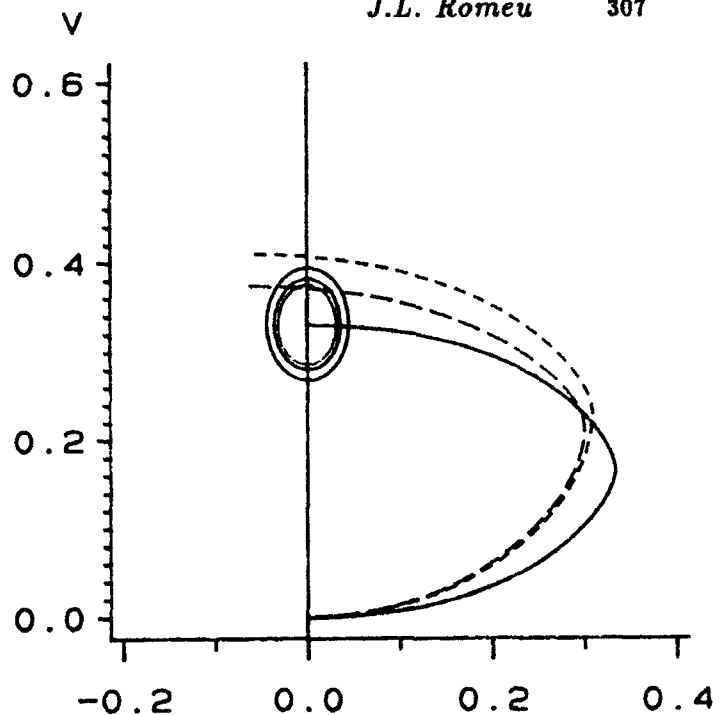


Figure 3.

4.0 Purely Kurtic Distributions.

In Figure 4 we show a sample from a bivariate Uniform, having skewness of 0.0 and kurtosis of less than 6.0 (purely kurtic).

We observe how the sample vector endpoint now falls way above the confidence ellipse, but on its vertical center axis. In addition, we can now observe a totally different vector pattern on the two halves of the linked vector path. The first half is much closer to the null pattern, crosses it and changes direction beyond the point where the null pattern does. Then, the upper half moves sharply upward.

This is a totally different pattern from the one presented in Figure 3, and allows us to recognize the sample as coming from a symmetric, flat, non normal distribution.

The pattern corresponding to a leptokurtic distribution is not presented for lack of space, but is also distinctive (Table 1).

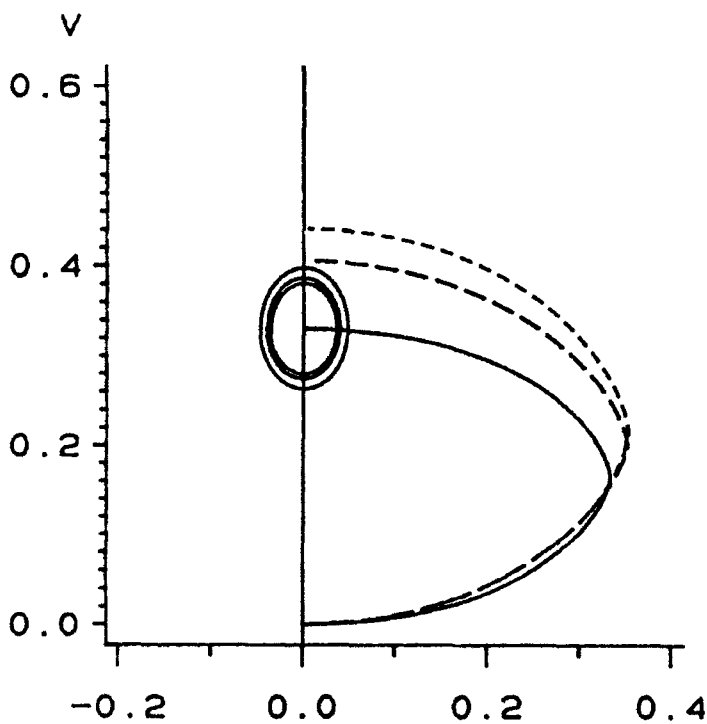


Figure 4.

5.0 Combined Skewed/Kurtic Distributions.

In Figure 5, we show a sample of pattern from a mixed skewed/kurtic distribution: the Bivariate Chi Square with 10 d.f.

We can see how the sample linked vector endpoint also falls off the confidence ellipse, entirely to its left. We can also notice how the vector pattern remains, during its entire trajectory, before (1st. half) or below (2nd half) the pattern of the null vector, which it never crosses. This characteristic pattern of the sample linked vector also provides a distinct test visualization.

It is with this graphical visualization that we recognize this sample as coming from a combined skewed and peaked non normal distribution.

6.0 Results.

On Table 1, we show a graphical comparison of six patterns of the sample linked vectors. The patterns have been subdivided into three parts: (i) lower and (ii) upper halves of the vector trajectory and (iii) endpoint position with respect to the confidence ellipse.

We have classified the six non normal distributions under study into three groups: (i) skewed, (ii) kurtic and (iii) combination. GLD-2 and GLD-3 are two bivariate distributions generated using the Generalized Lambda Distribution. Its objective is to provide, for comparison, an intermediate (skewed/kurtic) result between, respectively, the flat bivariate Uniform and the highly skewed GLD-1.

We can observe three distinctly different linked vector patterns, corresponding to these three non overlapping distribution classes. Hence, when rejecting multivariate normality, it is now also possible to identify the pattern of the sample and then to, (i) make an educated guess as to which (non normal) alternative

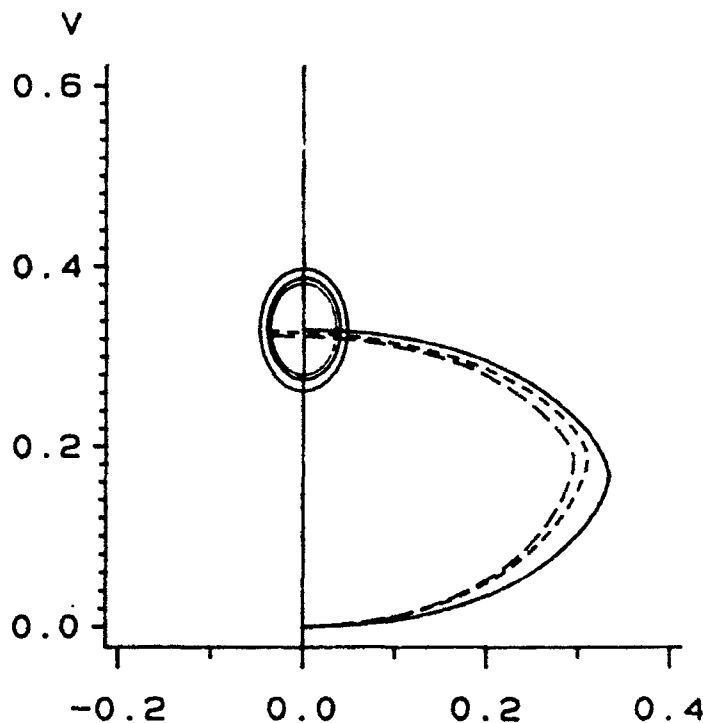


Figure 5.

distribution we want to test next. Or we may want to (ii) assess the type of non normal departure we are dealing with in order to implement the transformations necessary to redress the problems.




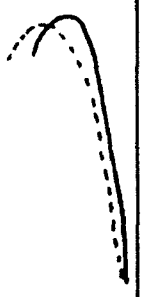
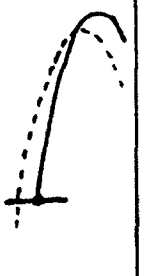
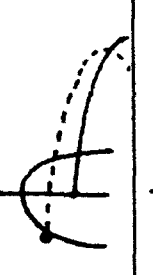
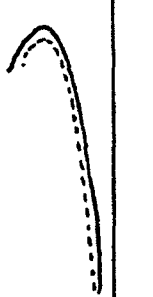

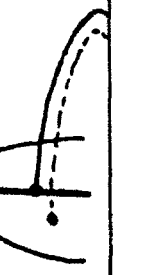

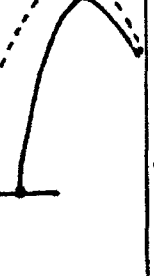
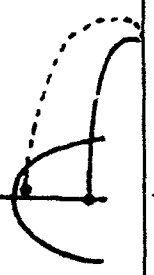

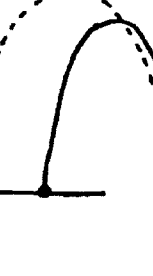
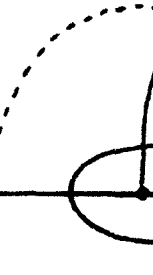

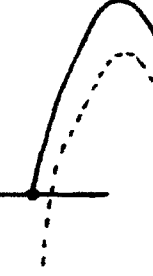
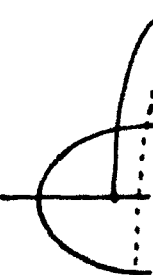
None of the other multivariate normality tests studied provides this capability.

6.0 Bibliography.

Ozturk, A. and J. L. Romeu (1992), A New Method for Assessing Multivariate Normality With Graphical Applications, *Communications in Statistics; Simula.*, 21(1), pages 15-34.

Romeu, J. L. (1990), Development and Evaluation of a General Procedure for Assessing Multivariate Normality, CASE Center Technical Report No. 9022, CASE Center of Syracuse University. Syracuse, NY. 13244-4100.

Table 1.
Vector Patterns:

Shape	Distrib.	1st. Half	2nd. half	Endpoint
Skewed	GLD - 1			
	GLD - 3			
Kurtic	t (8)			
	GLD - 2			
	UNIFORM			
Both	<u>Chi Square</u>			

Some Integration Strategies For Problems In Statistical Inference

Michael Evans
Department of Statistics
University of Toronto
Toronto, Ontario
Canada M5S 1A1

Tim Swartz
Department of Mathematics and Statistics
Simon Fraser University
Burnaby, British Columbia
Canada V5A 1S6

Abstract

This paper surveys the techniques and approaches available for the numerical approximation of integrals. There are many problems in statistics where the approximation of integrals is necessary. In particular implementing a Bayesian data analysis typically requires the evaluation of many integrals and the dimensionality of these can be very high. The study of Monte Carlo algorithms, suitable for such problems, is a research topic of great current interest in the statistical community.

1. Introduction

With many problems in statistical inference we are faced with the need to evaluate

$$\int_{R^k} f(\mathbf{x}) d\mathbf{x} \quad (1.1)$$

for some $f: R^k \rightarrow R$. In particular in Bayesian inference $f \geq 0$ is the product of the likelihood and prior and (1.1) is the normalizing constant for the posterior. We will denote the posterior density by p , namely p is f divided by (1.1). More generally, in the Bayesian context we want to evaluate the posterior expectation of $g: R^k \rightarrow R$, namely

$$\frac{\int_{R^k} g(\mathbf{x}) f(\mathbf{x}) d\mathbf{x}}{\int_{R^k} f(\mathbf{x}) d\mathbf{x}} \quad (1.2)$$

for many different g .

Sometimes highly accurate estimates of integrals are required in statistical contexts, e.g. the approximation of distribution functions for commonly used distributions. When these integrals depend on data produced by some random process, however, there is no point in pursuing a level of accuracy much beyond the level of variation in the value of the integral we would expect due to sampling error. In most Bayesian problems the latter situation obtains and hence even very high dimensional integrals can be realistically approximated.

In many statistical problems we often have asymptotic results which, while not necessarily accurate enough for

all situations, give us some guidance in the design of appropriate algorithms. For example Chen(1985) gives general conditions for the asymptotic normality of the posterior.

Also in Bayesian problems our primary interest may be in ratios or integrals as in (1.2). Some algorithms explicitly avoid the evaluation of (1.1). There are, however, many contexts where the normalizing constant (1.1) is required, e.g. assessing model adequacy and choosing a model.

These considerations give characteristics of integration problems in statistics which tend to distinguish them from the general kind of integration problem a numerical analyst might be confronted with. We now proceed to discuss four broad categories of integration techniques and their relevance to the evaluation of (1.1) and (1.2). We classify these as:

1. asymptotics
2. sampling from the posterior
3. multiple quadrature
4. importance sampling.

2. Asymptotics

Lindley(1961, 1980) proposed a normal approximation to the posterior density p . Chen(1985) gives general conditions for this approximation to apply. The approximation is

$$p(\mathbf{x}) \approx (2\pi)^{-k/2} (\det \hat{\Sigma})^{-1/2} \exp\left\{-\frac{1}{2}(\mathbf{x} - \hat{\mu})' \hat{\Sigma}^{-1}(\mathbf{x} - \hat{\mu})\right\} \quad (2.1)$$

where $\hat{\mu}$ is a local maximum of f and $\hat{\Sigma}$ is

$$(-\partial^2 \log f(\mathbf{x}) / \partial x_i \partial x_j)^{-1} \quad (2.2)$$

evaluated at $\hat{\mu}$. This leads to approximating (1.1) by

$$(2\pi)^{k/2} (\det \hat{\Sigma})^{1/2} f(\hat{\mu}). \quad (2.3)$$

If we can integrate g with respect to this multivariate normal then we have a natural approximation to (1.2). Alternatively we could expand g about $\hat{\mu}$ in a Taylor's series and integrate the first few terms of this expansion. Using only the first term in this expansion gives an $O(1/n)$ approximation when f is the unnormalized posterior and n is the sample size. If this is not feasible we can approximate (1.2) by generating X_1, \dots, X_N from the $N_k(\hat{\mu}, \hat{\Sigma})$ distribution and computing

$$\frac{1}{N} \sum_{i=1}^N g(X_i). \quad (2.4)$$

We note that (2.3) is Laplace's approximation to (1.1). Applying Laplace's method to both the numerator and denominator of (1.2), Tierney and Kadane(1986) gave an asymptotic approximation for this posterior expectation when g is nonnegative. Further they discussed the application of this technique to approximate the marginal density of the posterior distribution of g . This was extended to deal with nonpositive functions in Tierney, Kass and Kadane(1989). The advantage of using the Laplace technique for the numerator and denominator in (1.2) is an increase in the order of approximation to $O(1/n^2)$ without the need to compute derivatives beyond order 2.

Other features and applications of this approach to approximate Bayesian calculations have been discussed by Kadane, Kass and Tierney and other authors. These methods have been shown to give remarkable accuracy in a number of contexts and to be computationally efficient. One difficulty lies with how to assess the error in an approximation in a given context.

3. Sampling From The Posterior

Given the relatively low-level of accuracy demanded in many statistical situations, it can be argued that an ideal solution to integration problems in Bayesian inference would be an algorithm to generate a sample from the posterior density p . Thus we would generate X_1, \dots, X_N from p and approximate (1.2) by (2.4). By the Strong Law of Large Numbers the estimate converges almost surely to the correct value. If the posterior expectation of g^2 exists then the error in the approximation can be measured by the variance of (2.4) and N times this quantity is given by

$$\int_{R^k} g^2(x)p(x)dx - \left(\int_{R^k} g(x)p(x)dx\right)^2 \quad (3.1)$$

which is in turn estimated by

$$\frac{1}{N-1} \left[\sum_{i=1}^N g^2(X_i) - N \left(\frac{1}{N} \sum_{i=1}^N g(X_i) \right)^2 \right]. \quad (3.2)$$

There is a full discussion of the many algorithms which we might use for solving this problem in Devroye(1986). Of particular interest is the rejection algorithm due to von Neumann(1951). For this we need a density w satisfying $f \leq cw$, for some constant c and an algorithm for generating from w . We then generate X from w statistically independent of U generated from the Uniform(0,1) distribution. If $f(X) \geq cUw(X)$ then we accept X and it easy to show that in this case $X \sim p$. Suppose to obtain our sample X_1, \dots, X_N from p , we actually generated N^* values from w . Then, in addition to (2.4) as an estimate of (1.2), we have that cN/N^* estimates (1.1) in the sense that it converges almost surely to this value. The variance, times the Monte Carlo sample size N^* , of this estimate of (1.1) is given by

$$\left(\int_{R^k} f(x)dx \right) \left(c - \int_{R^k} f(x)dx \right) \quad (3.3)$$

which is estimated by

$$c^2 \frac{N}{N^*} \left(1 - \frac{N}{N^*} \right). \quad (3.4)$$

The difficulty in implementing the rejection algorithm, and the reason it does not generally provide a realistic solution to the problem, is that it is typically extremely difficult to find a w satisfying the inequality. Even when this is possible the resulting algorithm is often inefficient in the sense that most of the computing time is spent in computing generated values which are rejected. This difficulty increases as the dimension of the integral rises.

A current focus of interest in the statistical community is a class of algorithms which do not exactly generate a value from the posterior but only do so approximately with the approximation improving with the length of time the algorithm is run. As most of these algorithms are based on the probabilistic structure of a Markov chain, they are sometimes called Markov chain methods, see for example Tierney(1991). The basic idea is to construct a Markov chain X_1, X_2, \dots which has p as its unique stationary distribution. Then as we run the chain X_N converges in distribution to p as $N \rightarrow \infty$. Further, via an appropriate ergodic theorem, (2.4) converges almost surely to (1.2). Error assessment is primarily carried out by comparing estimates from runs of different lengths and inferring convergence when differences are small. Estimating (1.1) is not quite so straightforward but there are several possible strategies which we will not discuss here.

Markov chain methods are generally related to the Metropolis algorithm, the basic version of which was first presented in Metropolis, Rosenbluth, Teller and Teller(1953). For this we specify an initial, time homogeneous Markov chain on $\text{supp } f$, say with transition

density functions $q(\cdot | \mathbf{x})$ for each $\mathbf{x} \in \text{supp } f$ and an initial state $\mathbf{X}_0 \in \text{supp } f$. Then we generate a new Markov process as follows: given that we are in state \mathbf{X}_N at time N , generate \mathbf{Y} from $q(\cdot | \mathbf{X}_N)$ and put $\mathbf{X}_{N+1} = \mathbf{Y}$ with acceptance probability

$$A(\mathbf{X}_N, \mathbf{Y}) = \min\left\{\frac{f(\mathbf{Y})q(\mathbf{X}_N | \mathbf{Y})}{f(\mathbf{X}_N)q(\mathbf{Y} | \mathbf{X}_N)}, 1\right\} \quad (3.5)$$

else generate a new \mathbf{Y} . See Tierney(1991) for a discussion of what conditions are necessary for p to be the unique stationary distribution of this chain. A more general form of this algorithm can be given; see Gelman and Rubin (1992). The essential difficulty with this algorithm is an appropriate choice of q to ensure rapid convergence.

As a special case of the generalized Metropolis algorithm we have the Gibbs sampling algorithm. This was first presented in the context of a Bayesian approach to image restoration problems by Geman and Geman(1984). Its broader implications for Bayesian statistics were noted by Gelfand and Smith(1990). We describe a special case of this algorithm. Let $p_i(\cdot | X_1, \dots, X_{i-1}, X_{i+1}, \dots, X_k)$ denote the conditional posterior density of the i -th coordinate of \mathbf{X} given the remaining coordinates. The algorithm then proceeds as follows. We specify an initial state \mathbf{X}_0 . Then given the N -th state $\mathbf{X}_N = (X_{N,1}, \dots, X_{N,k})'$ we generate the $N+1$ -st state in k steps:

generate $X_{N+1,1} \sim p_1(\cdot | X_{N,2}, \dots, X_{N,k})$
 generate $X_{N+1,2} \sim p_2(\cdot | X_{N+1,1}, X_{N,3}, \dots, X_{N,k})$
 \vdots
 generate $X_{N+1,k} \sim p_k(\cdot | X_{N+1,1}, \dots, X_{N+1,k-1})$.

The convergence of this algorithm is discussed in Schervish and Carlin(1990). This can be very fast in many important practical problems. The convergence is affected, however, by the shape of the posterior and it can be very poor, see for example Evans, Guttman and Olkin(1991).

The Gibbs sampling algorithm described above requires that we have algorithms to generate from each of the conditional distributions. Hence implementation of the algorithm described above is restricted. This can be addressed in a number of ways. Sometimes latent variables can be added to the problem, increasing the dimension of the integral, in such a way that it is relatively straight-forward to generate from all of the conditionals of this augmented set of variables. This technique is due to Tanner and Wong(1987). If each of the conditional

densities is log-concave then Gilks and Wild(1992) have developed an adaptive rejection algorithm for generating from each of the conditionals. More generally Gelman and Rubin(1992) describe a generalized Metropolis algorithm which allows the conditionals to be replaced by approximations.

4. Multiple Quadrature

Multiple quadrature rules perhaps represent a more traditional approach to integration problems. While they are the technique of choice for relatively low dimension k , as k grows they become impractical and Monte Carlo methods become necessary. Research on these methods seems to be steadily increasing the dimension where they are suitable and may lead to practical methods for high-dimensional problems as well.

First we will discuss quadrature rules; i.e. rules for $k = 1$. A quadrature rule for approximating

$$\int_{-\infty}^{\infty} h(x)w(x)dx \quad (4.1)$$

where w is a density on R , takes the form

$$\sum_{i=1}^m w_i h(x_i) \quad (4.2)$$

for points x_i and weights w_i . For calculating (1.1) we put $h = f/w$ when $\text{supp } f \subseteq \text{supp } w$. For a thorough treatment of quadrature rules see Davis and Rabinowitz(1984).

There are many different classes of quadrature rules, determined by how we select the weights and points. Given that we have selected distinct points x_1, \dots, x_m a common method of determining the weights is to interpolate h at these points by a polynomial of degree $m-1$ and integrate this times w , assuming of course that w has its first $m-1$ moments. Such a rule is called an interpolatory rule and it obviously exactly integrates h whenever h is a polynomial of degree less than m . For example when w is the uniform density on some interval and the x_i are equispaced in the interval and include the end-points, we get the Newton-Cotes rules. When $m = 2$ this is the trapezoid rule and when $m = 3$ this is Simpson's rule. In an application these rules are often compounded; i.e. the interval is divided up into subintervals of equal length and the rule is applied in each subinterval. It can be shown that provided the rule integrates constants exactly then the compounded rule converges to (4.1) as the number of subintervals increases.

Given that we are free to choose the points, we might try to choose them optimally so that they exactly integrate all polynomials up to some maximal degree. Such

rules exist whenever w has all its moments up to order $2m$. They are called Gauss rules and the maximal degree is $2m - 1$. For example when w is the $N(0,1)$ density we get the Hermite rules, when w is a $\text{Gamma}(\alpha)$ density we get the Laguerre rules and when w is a $\text{Beta}(\alpha, \beta)$ density we get the Jacobi rules. Obviously Gauss rules have significance for many statistical applications. For a particular f the idea is to choose w so that $h = f/w$ is well-approximated by a low-degree polynomial and then use a Gauss rule for w to approximate (1.1). To assess the error in the approximation we repeat the calculation with a rule containing more points and compare the results. This is the usual method for assessing error when using multiple quadrature rules.

The simplest method of constructing multiple quadrature rules; i.e. a rule for $k > 1$, is to form product rules. Hence if

$$h(x_1, \dots, x_k) = \frac{f(x_1, \dots, x_k)}{w_1(x_1) \cdots w_k(x_k)} \quad (4.3)$$

where the w_i are densities, a product rule approximates (1.1) by

$$\sum_{i_1=1}^{m_1} \cdots \sum_{i_k=1}^{m_k} w_{1,i_1} \cdots w_{k,i_k} h(x_{1,i_1}, \dots, x_{k,i_k}) \quad (4.4)$$

where the x_{i,i_j} and w_{i,i_j} are the points and weights of a quadrature rule associated with w_i . Ignoring the difficulty of how to choose the w_i for the moment, we note that such rules suffer from a dimensional effect, namely to implement (4.4) requires $m_1 \cdots m_k$ function evaluations. For even modest dimensions, e.g. $k = 10$, this requires far too much computation to be practically useful. Further it can be shown that if we take all the rules to be the compounded trapezoid rule then the error in (4.4) is $O(1/N^{2/k})$ where N is the number of function evaluations. Hence it would appear that, since the rate of convergence of importance sampling, see the next section, is $O_P(1/N^{1/2})$ under fairly general conditions, then the product trapezoid rule is not competitive with Monte Carlo whenever $k > 4$. Similar results hold for other product rules.

Product rules can be effectively used, however, in lower dimensions. For example Naylor and Smith(1982) use product Hermite rules successfully in a number of problems. Their usage also illustrates another point concerning the application of multiple quadrature rules. For it is unlikely that f is well-approximated by a $N_k(0, I)$ density times a low degree polynomial. Asymptotics suggest, however, that this often will be the case if we replace the $N_k(0, I)$ density by a $N_k(\mu, \Sigma)$ density where μ is the posterior mean and Σ is the posterior variance

matrix. Hence Naylor and Smith(1982) use an adaptive approach which computes an initial approximation to the posterior means, variances and covariances, using product Hermite rules. They then transform the integrand using these quantities, so that the transformed posterior now has approximate mean 0 and variance matrix I . They then iterate this until the process stabilizes. Of course this may not work if the shape of the posterior is far from the normal but it makes the point that successful application of a multiple quadrature approach requires that the integrand be appropriately located, scaled and not have the bulk of its mass concentrated near a hyperplane.

Various attempts have been made to avoid the dimensional effect described above. One idea, due to Hammersley(1960), suggests that we estimate (4.4) by sampling the terms in this sum via Monte Carlo. When each rule in the product is a Gauss rule then $w_{i,i_j} > 0$ and $w_{i,i_1} + \cdots + w_{i,i_{m_i}} = 1$. Hence we can sample the terms of the sum using the weights as a discrete probability distribution. Evans and Swartz(1988) give an extensive analysis of this technique.

Another approach to avoiding the computational problems associated with product rules, is based on the fact that, when the i -th rule in the product rule integrates exactly all polynomials of degree n_i or less, then the product rule will exactly calculate

$$\int_{R^k} x_1^{i_1} \cdots x_k^{i_k} w(x) dx \quad (4.5)$$

where $w(x) = w_1(x_1) \cdots w_k(x_k)$ and $i_j \leq n_j$ for $j = 1, \dots, k$. We then look for rules with similar properties but requiring far fewer function evaluations. For a general density w on R^k a rule takes the form

$$\sum_{i=1}^m w_i h(x_i) \quad (4.6)$$

and if it calculates (4.5) exactly, whenever $i_1 + \cdots + i_k \leq d$ it is called a monomial rule of degree d . The problem then is to find a monomial rule of degree d with the smallest m . An important class of monomial rules is given by the fully symmetric rules. These arise when w is invariant under permutations and sign changes of the coordinates; e.g. the uniform density on $[-1, 1]^k$, and the x_i arise from applying these transformations to some finite set of points called the generators of the rule. There is one weight for each generator which is then also associated with each transformed point. Fully symmetric rules with smallest or close to smallest number of points have been obtained for a number of w . These are in general not easy problems to solve and sometimes the rules suffer from some of the points lying outside $\text{supp } w$.

Davis and Rabinowitz(1984) contains a more extensive discussion.

An interesting approach to generating multiple quadrature rules gets away from the requirement of exactly integrating monomials but constructs rules which will be good, in an average sense, for a class of integrands which arise from a stochastic process. This is sometimes called Bayesian quadrature as choosing the stochastic process is analogous to choosing a prior. Hence suppose w is a density and we want to calculate

$$\int_{R^k} h(x)w(x)dx \quad (4.7)$$

where h arises from a Gaussian process with some mean and covariance functions. The conditional distribution of (4.7) given the values of h at x_1, \dots, x_m is then normally distributed with mean and variance determined by the mean and covariance function of the process and the x_i . This conditional mean is of the form of a constant plus (4.6) and is a natural estimate of (4.7). The problem then is to choose these points to minimize the mean-squared error. This is discussed at some length in O'Hagan(1992). There are several problems which remain to be solved, not the least of which being the optimality problem, but the possibility exists that new and useful multiple quadrature rules will be obtained by this approach.

Lattice or quasirandom rules form another class of multiple quadrature rules. These typically require that the problem be transformed to

$$\int_{[0,1]^k} h(x)dx \quad (4.8)$$

where h is periodic; i.e. $h(x_1, \dots, x_{i-1}, 0, x_{i+1}, \dots, x_k) = h(x_1, \dots, x_{i-1}, 1, x_{i+1}, \dots, x_k)$. A lattice rule then takes the form of a specified sequence of points X_1, X_2, \dots in $[0, 1]^k$ and approximating (4.8) by

$$\frac{1}{N} \sum_{i=1}^N h(X_i) \quad (4.9)$$

for some N . Error assessment is performed by computing (4.9) for different N and comparing results.

When using such a rule the problem is to determine good sequences. To measure this there are numerous criteria; e.g. the worst-case discrepancy

$$D_N^{worst} = \sup_{t \in [0,1]^k} \left| \frac{1}{N} \sum_{i=1}^N I_{[0,t]}(X_i) - t_1 \dots t_k \right| \quad (4.10)$$

where $[0, t] = \prod_{i=1}^k [0, t_i]$, and the average-case discrepancy

ancy

$$D_N^{ave} = \left\{ \int_{[0,1]^k} \left[\frac{1}{N} \sum_{i=1}^N I_{[0,t]}(X_i) - t_1 \dots t_k \right]^2 dt \right\}^{1/2}. \quad (4.11)$$

For example the Hammersley sequence, see Davis and Rabinowitz(1984) for a definition of this, has been shown to satisfy $D_N^{worst} \leq O((\log N)^{k-1}/N)$ and hence has a better error rate than importance sampling. A recent result of Wozniakowski(1991) shows that a minor adjustment of the Hammersley sequence has an optimality property when using average-case complexity, $w = 1$ and h distributed as a Wiener process on $C[0, 1]^k$. For a discussion of the use of these rules in statistical contexts see Shaw(1988).

The subregion adaptive algorithms, as described in van Dooren and de Ridder(1976), Genz and Malik(1980) and Genz(1991), make use of multiple quadrature rules. For these algorithms the problem is transformed so that the domain of integration is $[0, 1]^k$. The algorithm proceeds iteratively as follows. Let $\epsilon > 0$ and start with one region $B_{11} = [0, 1]^k$. At the n -th step $[0, 1]^k$ has been partitioned into n subregions B_{n1}, \dots, B_{nn} and multiple quadrature rules have been applied in each subregion to get estimates R_{n1}, \dots, R_{nn} of the integrals of the integrand over these sets and also error estimates E_{n1}, \dots, E_{nn} . The error estimate E_{ni} is obtained by computing R_{ni} using rules of different orders and comparing the results. If $E_{n1} + \dots + E_{nn} < \epsilon$ the algorithm stops and otherwise splits the region B_{ni} with $E_{ni} = \max\{E_{nj} \mid 1 \leq j \leq n\}$. The partitioning algorithm takes the regions to be rectangles with sides parallel to the coordinate axes and uses a criterion to choose which side of the rectangle to split to form the subrectangles. The algorithm has been applied in some statistical contexts, see for example Genz and Kass(1991).

5. Importance Sampling

Suppose that w is a density on R^k satisfying $\text{supp } f \subseteq \text{supp } w$ and that we have an algorithm for generating samples from w . Let X_1, \dots, X_N be a sample from w . Then the importance sampling estimate of (1.1) is given by

$$\frac{1}{N} \sum_{i=1}^N \frac{f(X_i)}{w(X_i)} \quad (5.1)$$

and the estimate of the numerator of (1.2) is

$$\frac{1}{N} \sum_{i=1}^N \frac{g(X_i)f(X_i)}{w(X_i)}. \quad (5.2)$$

By the Strong Law of Large Numbers these quantities converge almost surely to the quantities they are estimating. Also the ratio of (5.2) to (5.1) converges to (1.2). We have that N times the variance of (5.1) is given by

$$\int \frac{f^2(x)}{w(x)} dx - \left(\int f(x) dx \right)^2 \quad (5.3)$$

and N times the variance of (5.2) is given by

$$\int \frac{g^2(x)f^2(x)}{w(x)} dx - \left(\int g(x)f(x) dx \right)^2 \quad (5.4)$$

and these integrals are implicitly over $\text{supp } w$. If both (5.3) and (5.4) are finite then the asymptotic variance of the ratio of (5.2) to (5.1) is

$$\frac{1}{N} \left(\frac{\mu_2}{\mu_1} \right)^2 \left[\left(\frac{\mu_1}{\sigma_1} \right)^2 + \left(\frac{\mu_2}{\sigma_2} \right)^2 - 2 \left(\frac{\mu_1}{\sigma_1} \right) \left(\frac{\mu_2}{\sigma_2} \right) \rho_{12} \right] \quad (5.5)$$

where μ_1 equals (1.1), σ_1^2 equals (5.3), μ_2 equals (1.2), σ_2^2 equals (5.4) and $\rho_{12} = \sigma_{12}/\sigma_1\sigma_2$ with

$$\sigma_{12} = \int \frac{g(x)f^2(x)}{w(x)} dx - \mu_1\mu_2. \quad (5.6)$$

Thus the variance of the importance sampling estimate of (1.2) is estimated by substituting the importance sampling estimates of each of the relevant quantities into (5.5). For example we estimate (5.3) by

$$\frac{1}{N-1} \left[\sum_{i=1}^N \left(\frac{f(X_i)}{w(X_i)} \right)^2 - N \left(\frac{1}{N} \sum_{i=1}^N \frac{f(X_i)}{w(X_i)} \right)^2 \right]. \quad (5.7)$$

Of course we want to choose w to ensure finite variance of our estimates. If we can choose w such that $f \leq cw$ for some constant c then the importance sampling estimate of (1.1) has finite variance. Due to the complexity of many of the integrands which arise in Bayesian inference this is often very difficult to guarantee. Hence we should be conservative and select a w we feel has a good chance of achieving a finite variance for all the quantities we want to estimate. When the above inequality is achievable a direct comparison between (5.3) and (3.3) shows that importance sampling is always at least as accurate as rejection sampling and hence is the preferred algorithm in this context.

The basic problem in using importance sampling is in determining methods for choosing an appropriate w . In a Bayesian problem we will want a w which leads to good estimators of (1.2) for many different g . Typically, however, f is the dominating influence in the integrand gf and so this is achieved when we have w suitable for estimating (1.1). For this problem it is easy to show that

the optimal choice, in that it minimizes (5.3), is $w \propto f$. This is of course unrealistic, but it underscores the point that the success of importance sampling depends on the closeness of w to p , the normalized version of f , and $\int (f^2(x)/w(x)) dx$ provides a suitable measure of this.

We now discuss methods of selecting a suitable w . For this let $\mathcal{W} = \{w_\lambda \mid \lambda \in \Lambda\}$ be a class of possible importance samplers for a problem. For example, and this is a commonly used class, we could take \mathcal{W} to be the class of distributions generated by $\mathbf{a} + B\mathbf{Z}$ where $\mathbf{a} \in R^k$, $B \in R^{k \times k}$ is lower triangular with positive diagonal elements and \mathbf{Z} is distributed as a k -dimensional Student(τ) distribution. Hence in this case $\lambda = (\mathbf{a}, B, \tau)$. Naturally we would like to choose $\lambda \in \Lambda$ to minimize $\int (f^2(x)/w_\lambda(x)) dx$. This is generally a difficult problem to solve but it may be worthwhile computationally to do so, particularly if we are going to use w to compute many posterior expectations.

Given the difficulty of solving the optimality problem there is a natural alternative, namely choose a $w \in \mathcal{W}$ which agrees with p in some set of selected characteristics. We call this matching characteristics. For example when using the Student family discussed above, a natural choice is to take \mathbf{a} to be the mode of the distribution and B to be the Cholesky factor of (2.2) evaluated at the mode. Of course this requires the unimodality of f . The choice of τ is more difficult but as the finiteness of the variance of our estimators is determined by the tail-length of the importance sampler, it is best to be conservative and choose τ low.

Using the Student family means that we are implicitly relying on the posterior having roughly elliptical contours. There are many examples where this is not the case and hence we would like families of importance samplers which exhibit a wider variety of shapes. Geweke(1989) generalizes the Student family to the split-Student family and provides a fitting algorithm. This allows for some skewness in the distribution but at the cost of a discontinuity in the density.

An alternative approach to choosing $w \in \mathcal{W}$ by matching characteristics is to do this via adaptive importance sampling. For this we select a vector of characteristics $\mathbf{m}(\lambda) \in R^l$ for w_λ , each of which can be expressed as an expectation; e.g. moments, probability contents, etc.. Let \mathbf{m}_* be the corresponding characteristics for the posterior p . Now select an initial importance sampler $w_{\lambda_0} \in \mathcal{W}$. Then generate a sample $\mathbf{X}_1, \dots, \mathbf{X}_N$ from w_{λ_0} and estimate the components of \mathbf{m}_* using the importance sampling estimates. Denote these estimates by $\mathbf{m}_{*,1}$. Next find $\lambda_1 \in \mathcal{W}$ which minimizes $\|\mathbf{m}_{*,1} - \mathbf{m}(\lambda)\|$. Then generate $\mathbf{X}_{N+1}, \dots, \mathbf{X}_{2N}$ from w_{λ_1} and combine this with the previous sample to get a new estimate

$m_{.2}$. For example if the mean of the first coordinate of $p(x_1, \dots, x_k)$ is an element of $m_{.2}$, then one way of combining to get a new estimate is

$$\frac{\sum_{i=1}^N X_{i1} f(X_i) / w_{\lambda_0}(X_i) + \sum_{i=N+1}^{2N} X_{i1} f(X_i) / w_{\lambda_1}(X_i)}{\sum_{i=1}^N f(X_i) / w_{\lambda_0}(X_i) + \sum_{i=N+1}^{2N} f(X_i) / w_{\lambda_1}(X_i)} \quad (5.8)$$

Then find $\lambda_2 \in \mathcal{W}$ minimizing $\|m_{.2} - m(\lambda)\|$. This can be run for a finite number of steps to select an importance sampler w to use to compute the remaining integrals of interest. Algorithms related to the above are discussed in Smith et. al.(1987), Evans(1991a), Oh(1991) and Oh and Berger(1991). In particular Oh and Berger(1991) prove a convergence result, under certain conditions, when we allow infinitely many steps. West(1990) discusses another approach to adaptive importance sampling using mixtures of multivariate Student distributions. The technique of chaining is developed in Evans(1991b) to aid in the selection of $w_{\lambda_0} \in \mathcal{W}$ as the algorithm can be sensitive to this choice.

When a given importance sampler has been selected there are numerous variance reduction techniques that can be used to increase the accuracy of the estimates. Primary amongst these are the use of control variates, stratified sampling and systematic sampling or, as it is sometimes called, antithetic variates. We refer to Hammersley and Handscomb(1964) for a discussion of these.

6. References

- Chen, C-F. (1985). On the asymptotic normality of limiting density functions with Bayesian implications. *JRSSB*, 47, No. 3, 540-546.
- Davis, P.J. and Rabinowitz, P. (1984). *Methods of Numerical Integration*, Second Edition. Academic Press.
- van Dooren, P. and de Ridder, L. (1976). Algorithm 6. An adaptive algorithm for numerical integration over an n-dimensional cube. *J. Comput. Appl. Math.*, 2, 207-217.
- Devroye, L. (1986). *Non-Uniform Random Variate Generation*. Springer-Verlag.
- Evans, M. (1991a). Adaptive importance sampling and chaining. *Statistical Multiple Integration*, eds. N. Flournoy and R.K. Tsutakawa, Contemporary Mathematics, 115, 137-143, Amer. Math. Soc..
- Evans, M. (1991b). Chaining via annealing. *Ann. Statist.*, Vol. 19, No. 1, 382-393.
- Evans, M., Guttman, I. and Olkin, I. (1991). Numerical aspects in estimating the parameters of a mixture of normal distributions. *Technical Report No. 9117*, Dept. of Statistics, University of Toronto.
- Evans, M. and Swartz, T. (1988). Sampling from Gauss rules. *SIAM J. Sci. Stat. Comput.*, Vol. 9, No. 5, 950-961.
- Gelfand, A.E. and Smith, A.F.M. (1990). Sampling based approaches to calculating marginal densities. *JASA*, 85, 398-409.
- Gelman, A. and Rubin, D.B. (1992). Honest inferences from iterative simulation. *Manuscript*.
- Geman, S. and Geman, D. (1984). Stochastic relaxation, Gibbs distributions and the Bayesian restoration of images. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 6, 721-741.
- Genz, A. (1991). Subregion adaptive algorithms for multiple integrals. *Statistical Multiple Integration*, eds. N. Flournoy and R.K. Tsutakawa, Contemporary Mathematics, 115, 23-31, Amer. Math. Soc..
- Genz, A. and Kass, R. (1991). An application of subregion adaptive numerical integration to a Bayesian inference problem. *Technical Report No. 523*, Dept. of Statistics, Carnegie Mellon University.
- Genz, A. and Mallik, A. (1980) An adaptive algorithm for numerical integration over an n-dimensional rectangular region. *J. Comp. Appl. Math.*, 6, 295-302.
- Geweke, J. (1989). Bayesian inference in econometric models using Monte Carlo integration. *Econometrica*, 57, 1317-1339.
- Gilks, W.R. and Wild, P. (1992) Adaptive rejection sampling for Gibbs sampling. *Appl. Stat.*, to appear.
- Hammersley, J.M. (1960). Monte Carlo methods for solving multivariable problems. *Ann. New York Acad. Sci.*, 86, 844-874.
- Hammersley, J.M. and Handscomb, D.C. (1964). *Monte Carlo Methods*. Methuen and Co.
- Lindley, D.V. (1961). The use of prior probability distributions in statistical inference and decisions. *Proc. 4th Berkeley Symp.* 1, 453-468.

- Lindley, D.V. (1980). Approximate Bayesian methods. *Bayesian Statistics*, eds. J.M. Bernardo, M.H. De Groot, D.V. Lindley and A.F.M. Smith. Valencia, Spain : University Press.
- Metropolis, N., Rosenbluth, A.W., Teller, A.H., and Teller, E. (1953). Equations of state calculations by fast computing machines. *J. Chemical Physics*, 21, 1087-1091.
- Naylor, J.C. and Smith, A.F.M. (1982). Applications of a method for the efficient computation of posterior distributions. *Appl. Stat.*, 31, 214-225.
- von Neumann, J. (1951). Various techniques in connection with random digits. *NBS Appl. Math. Ser.*, 12, 36-38.
- Oh, M-S. (1991). Monte Carlo integration via importance sampling : dimensionality effect and an adaptive algorithm. *Statistical Multiple Integration*, eds. N. Flournoy and R.K. Tsutakawa, Contemporary Mathematics, 115, 165-187, Amer. Math. Soc..
- Oh, M-S. and Berger, J. (1992). Adaptive importance sampling in Monte Carlo integration. *J. Stat. Comp. and Simul.* to appear.
- O'Hagan, A. (1991). Bayes-Hermite integration. *J. Statist. Plan. Inf.*, 29, 245-260.
- Schervish, M. and Carlin, B.P. (1991). On the convergence of successive substitution sampling. *Manuscript*.
- Shaw, J.E.H. (1988). A quasirandom approach to integration in Bayesian statistics. *Ann. Statist.*, Vol 16, No. 2, 895-914.
- Smith, A.F.M., Skene, A.M., Shaw, J.E.H. and Naylor, J.C. (1987). Progress with numerical and graphical methods for practical Bayesian statistics. *The Statistician*, 36, 75-82.
- Tanner, M.A. and Wong, W.H. (1987). The calculation of posterior distributions by data augmentation. *JASA*, 82, 528-540.
- Tierney, L. and Kadane, J.B. (1986). Accurate approximations for posterior moments and marginal densities. *JASA*, Vol. 81, No. 393, 82-86.
- Tierney, L., Kass, R. and Kadane, J.B. (1989). Fully exponential Laplace approximations to expectations and variances of nonpositive functions. *JASA*, 84, 710-716.
- Tierney, L. (1991). Markov chains for exploring posterior distributions. *Technical Report No. 560*, School of Statistics, University of Minnesota.
- West, M. (1992) Bayesian computations : Monte Carlo density estimation. *JRSSB* to appear.
- Wozniakowski, H. (1991). Average case complexity of multivariate integration. *Bull. AMS*, Vol. 24, No. 1, 185-194.

Posterior Integration in Dynamic Models

Peter Mueller

Institute of Statistics & Decision Sciences

Duke University

Durham, NC 27706

Abstract

The analysis of general dynamic models involves a sequence of posterior distributions corresponding to the subsequent stages of the dynamic model. In the absence of normal/linear structure numerical integration schemes are required to estimate features of these posterior distributions.

This paper reviews some previously suggested Monte Carlo based algorithms and suggests a new scheme which makes use of a Metropolis type algorithm to propagate a Monte Carlo sample simulated from the initial prior distribution through all stages of the dynamic model. For each of the posterior distributions in the dynamic model, the algorithm makes a Monte Carlo sample available which allows then to estimate posterior integrals as desired.

Before proceeding to the analysis at time t , the algorithm requires reconstruction of the posterior distribution corresponding to period $t - 1$. This is solved by an implementation of a mixture of Dirichlet process model, making use of the already available Monte Carlo sample.

1 Introduction

A general dynamic model is defined by an observation equation which specifies the sampling distribution for the observations y_t conditional on a parameter vector θ_t at time t and an evolution equation which determines the change of the unknown parameter vector between the stages of the dynamic model.

Observation equation: $y_t \sim p(y_t|\theta_t)$,

Evolution equation: $\theta_t = g_t(\theta_{t-1}) + \omega_t$.

Here $p(y|\theta)$ is a known, but not necessarily normal, conditional p.d.f., $g(\theta)$ is a known, possibly non-linear function and ω_t is a random vector with arbitrary, possibly non-normal distribution. See Pole and West (1988, 1990) or West and Harrison (1989) for more discussion of the general dynamic model.

Except for special cases like a normal/linear model and extensions thereof Bayesian analysis of general dynamic models leads to analytically intractable posterior integrals, with the additional difficulty that we have to deal with a whole sequence of posteriors $\pi(\theta_t|D_t)$ and $\pi(\theta_t|D_{t-1})$, corresponding to the consecutive stages of the dynamic model. Here $D_t = \{y_1, \dots, y_t\}$ is the set of all data up to time t . The posteriors $\pi(\theta_t|D_t)$ and $\pi(\theta_t|D_{t-1})$ reflect the information about the parameter vector θ_t in light of all the observations up to time t and $t - 1$ respectively.

A complicating feature of these models is that in general it is not even possible to give closed form expressions for the posteriors $\pi(\theta_{t+1}|D_t)$. Before advancing to inference at time $t + 1$, it is therefore necessary to somehow reconstruct $\pi(\theta_{t+1}|D_t)$. Previously suggested solutions include the use of normal/linear approximations, splines, simple multivariate normal approximations and use of mixture of multivariate t distributions (Pole and West 1988, Pole and West 1990, West and Harrison 1989, West 1990b and Mueller 1991).

2 Monte Carlo Integraion in General Dynamic Models.

2.1 Outline

In Mueller (1991) a Monte Carlo integration scheme for general dynamic models was suggested which pushes a simulated Monte Carlo sample from the initial prior distribution through all the steps of the dynamic model, making for each step a Monte Carlo sample from the current posterior available.

(i) The algorithm starts with a Monte Carlo sample $A_1 = \{\theta_1, \dots, \theta_m\}$, simulated from the original prior distribution $\pi(\theta_1|D_0)$.

(ii) Using a Metropolis type of algorithm this initial prior sample A_1 is transformed into a Monte Carlo sample $B_1 = \{\eta_1, \dots, \eta_m\}$ from $\pi(\theta_1|D_1)$. Details of this step are explained in Section 2.2.

(iii) By direct simulation of the evolution equation B_1 is transformed into a Monte Carlo sample $A_2 = \{\theta_1, \dots, \theta_m\}$ from $\pi(\theta_2|D_1)$. This is done by generating ω_i from the distribution given in the evolution equation and setting $\theta_i = g_1(\eta_i) + \omega_i$.

Continuing in an inductive way eventually provides a Monte Carlo sample from each of the relevant posteriors $\pi(\theta_i|D_i)$. These Monte Carlo samples can then be used to estimate posterior integrals as desired.

Figures 1 and 2 illustrate the outlined scheme. The two figures depict the posterior distributions $\pi(\theta_{39}|D_{38})$, $\pi(\theta_{39}|D_{39})$ and $\pi(\theta_{40}|D_{39})$ for two parameters in a dynamic model. The points in Figure 1 represent a Monte Carlo sample from $\pi(\theta_{39}|D_{38})$, the set of contours centered more towards the right represents a reconstruction of this posterior. The incoming observation y_{39} shifts the posterior distribution towards smaller values of the first parameter. The contours of $\pi(\theta_{39}|D_{39})$ are the concentric ellipses on the left of Figure 1. Adding the evolution noise slightly flattens the posterior again. Figure 2 shows a Monte Carlo sample and contours for $\pi(\theta_{40}|D_{39})$.

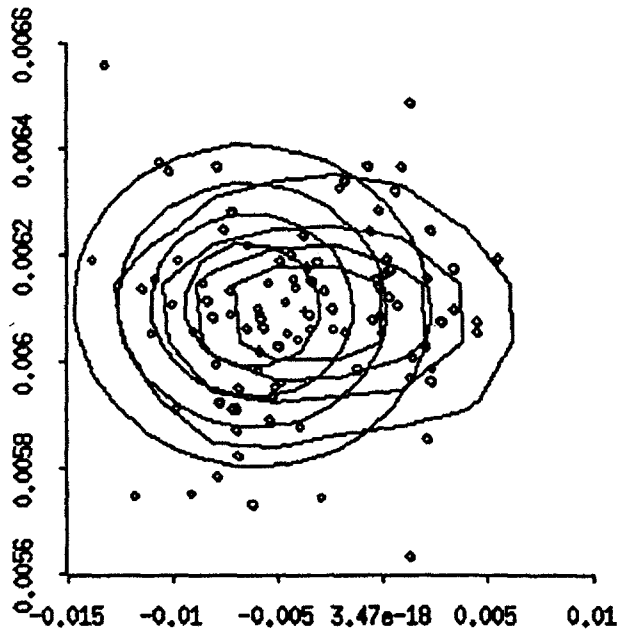


Figure 1: MC sample from $p(\theta^{(39)}|D_{38})$ (contours centered more on the right) with contours for $p(\theta^{(39)}|D_{39})$ (contours more on the left).

The weak point of this scheme is that the application of the Metropolis algorithm in Step (ii) will require the reconstruction of the density $\pi(\theta_i|D_{i-1})$ from the available Monte Carlo sample A_i . In Section 2.3 an application of

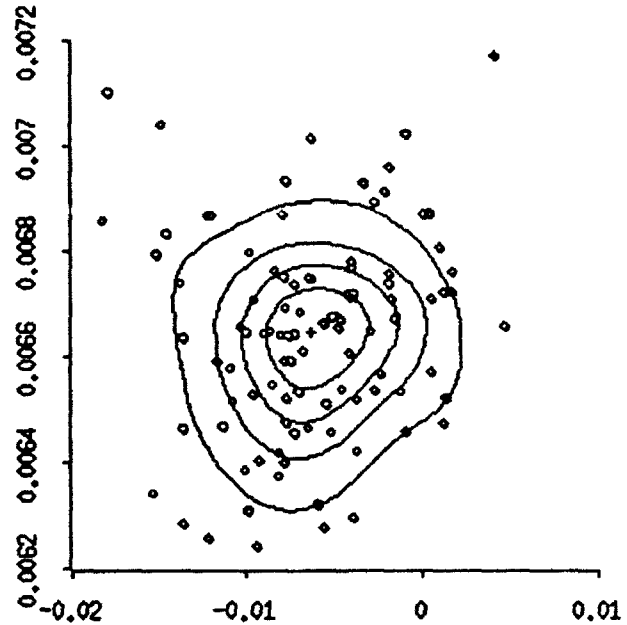


Figure 2: MC sample from $p(\theta^{(40)}|D_{39})$.

a Bayesian density estimation model will be suggested to derive a reconstruction $\hat{\pi}(\theta_i|D_{i-1})$.

2.2 Metropolis Algorithm

With respect to the isolated experiment of observing y_t , the distributions $\pi(\theta_t|D_{t-1})$ and $\pi(\theta_t|D_t)$ play the roles of prior and posterior. Transforming the Monte Carlo sample from $\pi(\theta_t|D_{t-1})$ into a Monte Carlo sample from $\pi(\theta_t|D_t)$ corresponds thus to "simulating" Bayes theorem for this experiment. While this is difficult in a general setting, it is facilitated in the dynamic model context by the fact that the additional observation y_t typically does not dramatically shift the posterior, i.e. $\pi(\theta_t|D_{t-1})$ can in general be expected not to be too far away from $\pi(\theta_t|D_t)$.

This feature makes application of the following algorithm attractive. Starting with a Monte Carlo sample point from $\pi(\theta_t|D_{t-1})$ simulate a Markov chain, $x_k, k = 0, 1, \dots, M$, which is defined such that $\pi(\theta_t|D_t)$ is asymptotic distribution. Since the initial distribution is already close to the asymptotic distribution, only a few simulated iterations suffice to obtain an approximate draw x_M from $\pi(\theta_t|D_t)$. An easily implemented Markov chain is given by the Metropolis algorithm described in Metropolis et al. (1953) and Hastings (1970). Tierney (1991a and 1991b) discusses the application of this algorithm and extensions to exploration and simulation of posterior distributions.

Let $p(\cdot)$ denote $\pi(\theta_t|D_t)$. Starting with a sample point

θ_i from $\pi(\theta_i|D_{i-1})$ simulate the following Markov chain x_k .

- (i) Start with $x_0 = \theta_i$.
- (ii) Generate a "candidate" $y \sim g(y|x_0)$.
- (iii) With probability $\alpha(x_0, y) = \min(1, p(y)/p(x_0))$ move to y , i.e. $x_1 := y$, else $x_1 = x_0$.
- (iv) Repeat to generate x_2, x_3 etc. and stop after M iterations. Take x_M as approximate sample point from $p(\cdot) = p(\theta^{(i)}|D_i)$.

The "probing distribution" $g(y|x)$ in (ii) is an arbitrary conditional distribution, chosen with the only restriction that $g(y|x) = g(x|y)$. For example $g(y|x)$ might be specified as $N(x, \Sigma)$ with Σ an estimate of the covariance matrix for $\pi(\theta_i|D_i)$. The rationale is that the candidates should preferably be generated in directions of high posterior probability.

The drawback of simulating the transition from $\pi(\theta_i|D_{i-1})$ to $\pi(\theta_i|D_i)$ by this scheme is that it requires pointwise evaluation of the posterior $\pi(\theta_i|D_i)$ when evaluating the acceptance probabilities $\alpha(x_k, y)$.

2.3 Mixture of Dirichlet Process

A problem particular to posterior integration in dynamic models is that the target distribution is in general not even available for pointwise evaluation. Following the scheme outlined in the earlier part of this paper however, there would be an available Monte Carlo sample A_i from $\pi(\theta_i|D_{i-1})$. If A_i could be used to derive a reconstruction $\hat{\pi}(\theta_i|D_{i-1})$, we could evaluate $\pi(\theta_i|D_i)$ by $\hat{\pi}(\theta_i|D_i) \propto \hat{\pi}(\theta_i|D_{i-1})p(y_i|\theta_i)$. Pointwise evaluation of the posterior $\pi(\theta_i|D_i)$ would then be enough to apply the Metropolis algorithm described in Section 2.2., thus completing the scheme outlined in 2.1.

A convenient, because technically straightforward, way to reconstruct the unknown density from the available Monte Carlo sample is given by an application of the mixture of Dirichlet process (MDP).

The MDP model gives a framework for Bayesian density estimation of an unknown p.d.f. $p(x)$, given a sample $\{x_i, i = 1, \dots, n\}$ from $p(x)$.

Assume that $x_i = \mu_i + \epsilon_i$ where $\mu_i \sim G$, with G drawn from a Dirichlet process $D(G_0, \alpha)$ with base measure G_0 and concentration parameter α . The ϵ_i are assumed i.i.d. $N(0, V)$. If the base measure G_0 of the Dirichlet process is assumed to be a normal distribution as well, then it can be shown that the posterior predictive distribution $\hat{p}(x)$ for an additional draw $x = x_{n+1}$ is a mixture of normals, respectively a mixture of Student-t's when the covariance matrix V is allowed to be random. The estimate $\hat{p}(x)$ will be a combination of the a priori predictive distribution which is determined by G_0 and point masses spread out around the estimated μ_i 's. From a practical point of

view, using the predictive density \hat{p} from an MDP model as density estimate is very similar to using a conventional kernel density estimate. West (1990) discusses this parallel.

For a full discussion of the MDP model see Antoniak (1974), Ferguson (1973), Escobar (1988) and Escobar and West (1990). Escobar and West also give an efficient and easily implemented Gibbs sampling scheme to estimate the MDP model. Although often only formulated in terms of univariate distributions, the MDP model is without modifications applicable for multivariate density estimation as well.

Figures 3 and 4 illustrate the flexibility of the MDP model for density reconstruction. The first figure shows a Monte Carlo sample with some of the sample points clustering around what might be secondary mode. The MDP reconstruction picks up on this feature and estimates a second mode. The other figure shows a Monte Carlo sample from a bivariate normal distribution truncated to be between 0 and 1. While the implemented MDP model does not allow for truncated distributions, it mimics the truncation quite closely by using an appropriate mixture of normals.

Both these situations can become important in the analysis of dynamic models. Heavy tails, common in some applications, easily lead to multimodality. An example is the model in Section 4. Appropriate modelling of constraints on the parameters becomes important in high dimensional parameter spaces when simple discarding becomes problematic.

3 Monte Carlo integration with MDP density reconstruction

Combining the Metropolis type simulation of Bayes theorem and the MDP reconstruction with the basic Monte Carlo scheme outlined in Section 2.1. leads then to the following algorithm.

- (i) Start with a Monte Carlo sample $A_1 = \{\theta_1, \dots, \theta_m\}$ from the initial prior distribution $\pi(\theta_1|D_0)$.
- (ii) Assume a Monte Carlo sample $A_i = \{\theta_i, i = 1, \dots, m\}$ from $\pi(\theta_i|D_{i-1})$ is available. Use the sample A_i to estimate a MDP model for $\pi(\theta_i|D_{i-1})$, giving an approximation $\hat{\pi}(\theta_i|D_{i-1})$.
- (iii) Using $\hat{\pi}(\theta_i|D_{i-1})$ proceed according to the Metropolis type algorithm outlined in Section 2.2. This will transform A_i into a Monte Carlo sample $B_i = \{\eta_1, \dots, \eta_m\}$ from $\pi(\theta_i|D_i)$.
- (iv) By simulating the evolution equation transform B_i into A_{i+1} . Use the Monte Carlo samples B_i and A_{i+1} to estimate posterior integrals as desired, including param-

eter estimates, forecasts and more.

4 Example: Exchange rate data

The statistical properties of exchange rate data have attracted considerable interest in the literature. The problem with analyzing exchange rate data is best highlighted by Meese and Rogoff's (1988) paper. They compare a variety of structural and time series models with a simple random walk model, and conclude that the random walk model performed best amongst all the competing models in terms of out-of-sample root mean squared forecasting error.

Friedman and Vandersteel (1982) argue for a normal sampling distribution with time-varying mean and variance parameters to model the often noticed leptokurtosis of exchange rate data. Hsieh (1988) and Diebold (1988) have modeled time-varying variance by versions of the ARCH (autoregressive conditional heteroskedasticity) model introduced by Engle (1982). Quintana and West (1987) suggested a dynamic multivariate Bayesian model.

The model which we consider in this paper combines elements of both, the ARCH model and the dynamic multivariate model. The univariate ARCH model is extended by allowing covariances between the time series and introducing dynamic parameters. Motivated by the discussion in the literature about the leptokurtosis of exchange rate data a multivariate Student *t* sampling distribution is chosen. The data consists of daily observations on the exchange rates for US dollar (D_t), German mark (M_t) and French franc (F_t) quoted in terms of Swiss franken. The data has been transformed to log differences.

Observation:

$$\begin{aligned} (D_t \ M_t \ F_t)' &= (\mu_t^D \ \mu_t^M \ \mu_t^F)' + e_t, \\ e_t &\sim \text{MV-t}(0, \Sigma_t; \nu), \end{aligned}$$

Evolution:

$$\begin{aligned} \Sigma_t^* &= C_t \Sigma_t C_t + B_t Q_t B_t, \\ Q_t &= [x_{t-q}, \dots, x_t] \cdot R_t \cdot [x_{t-q}, \dots, x_t]', \\ \theta_{t+1} &= \theta_t^* + \omega_t, \\ \omega_t &\sim N(0, \Omega_t), \end{aligned}$$

where $\theta_t = \text{vec}(\mu_t, \Sigma_t, B_t, R_t)$ is the parameter vector at time t , which contains the coefficients $(\mu_t^D, \mu_t^M, \mu_t^F)$, the elements of the observation covariance matrix Σ_t , and the ARCH coefficients R_t, B_t and C_t (θ_t^* replaces Σ_t by Σ_t^*). To reduce the dimensionality of the parameter vector the matrices R_t, B_t and C_t are modelled in terms of only two scalar parameters b_t and r_t : $B_t = \text{diag}(\sqrt{b_t}, \sqrt{b_t}, \sqrt{b_t})$ and $C_t = \max(0, I - B_t)$. The matrix R_t is given by

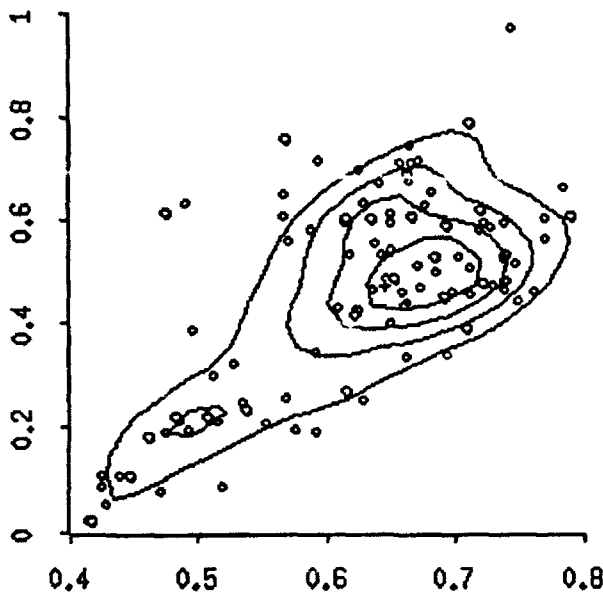


Figure 3: MC sample from $p(b_{87}, \rho_{87} | D_{86})$ in the exchange rate example of Section 4.

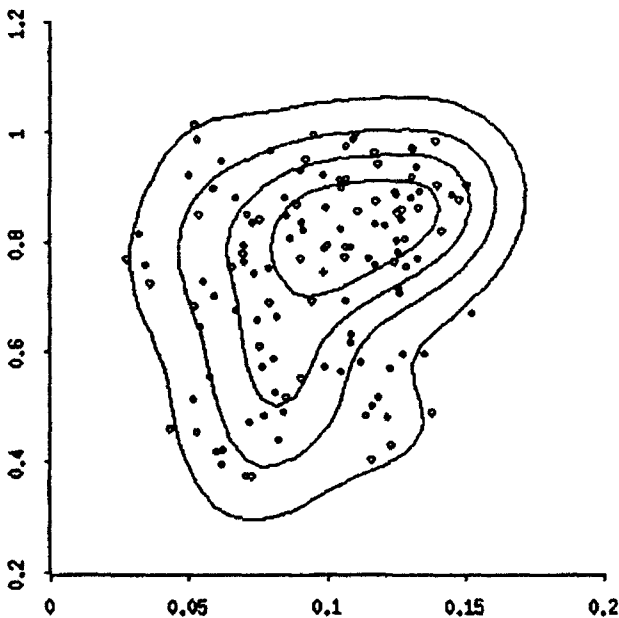


Figure 4: MC sample from $p(\alpha_1, \beta_1 | D_0)$ in the advertising awareness example of Migon and Harrison (1985).

$R_t = \text{diag}(w_1, \dots, w_q)$ with linearly increasing weights w_i
 $w_i = (1 - r_t)/q + 2 \cdot i/(q^2 + q) \cdot r_t$.

Both parameters, b_t and r_t , are restricted between 0 and 1.

If $b_t = 0$ then the ARCH component of the model drops out and the evolution for Σ_t reduces to a random walk. On the other extreme, if $b_t = 1$ then Σ_t^* would be modelled only in terms of the most recent q observations.

The interpretation of r_t is as a "down-weighting" parameter. If $r_t = 0$, then all q previous observations x_{t-i} , $i = 0, \dots, (q-1)$ enter with the same weight $1/q$ into the residual matrix Q_t of the evolution equation. On the other extreme, $r_t = 1$ implies that the observations enter with linearly increasing weights, starting with weight 0 for x_{t-40} . The decreasing ARCH coefficients w_i are consistent with the intuition that high volatility in the distant past has less of an effect on current volatility than high volatility in immediately preceding periods. Diebold (1988) considered linear and geometric models for the w_i 's and found strong evidence in favor of the linear law.

The covariance matrix of the evolution noise was chosen using the concept of discount factors as $\Omega_t = \delta W_t$, where W_t is the posterior covariance matrix of the full parameter vector at time t . See West and Harrison (1989) for more discussion of the idea of using discount factors to specify evolution noise. We chose $\delta = 0.09$.

The lag length q for the ARCH equation was set to $q = 40$. The degree of freedom ν for the multivariate t sampling distribution was chosen $\nu = 2$.

The initial prior specification is guided by the results of the studies of Hsieh (1988) and Diebold (1988). The μ_t parameters are all given a zero-mean prior with standard deviations corresponding to the range of parameter estimates obtained by Hsieh and Diebold. The prior on the covariance matrix $p(\Sigma_1|I_0)$ is modeled with a Wishart prior distribution with mean corresponding to a diagonal matrix with $\sigma^2 = 1.69 \cdot 10^{-6}$ in the diagonal and widest possible variance (i.e. degrees of freedom of the $p = 3$ dimensional Wishart are chosen as $\nu = p + 1$). The ARCH parameters r_1 and b_1 are specified with a normal $N(0.6, 0.2^2)$ prior for r_1 and a $N(0.8, 0.2^2)$ prior for b_1 . In the context of a dynamic model with discounting, the initial prior specification has importance only for the first few periods and is quickly washed out by the evolution noise ω_t and the - over many periods - relatively strong information from the data.

Figure 5 plots the three time series as log exchange rates relative to 1979. Figures 6 and 7 plot the trajectories for the variance and correlation parameters. The high correlation between franc and mark towards the later part of the time series reflect the increasingly closer co-operation within the European monetary system. The

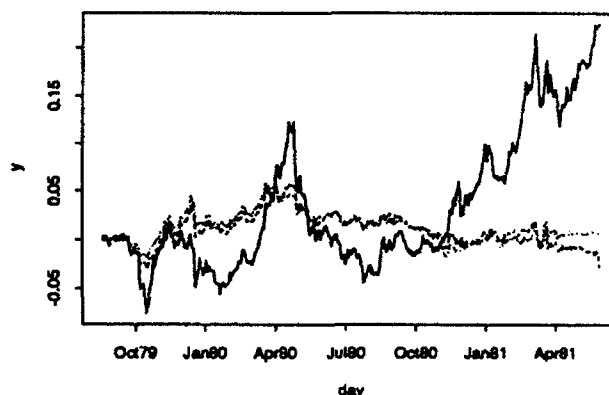


Figure 5: Log exchange rates for dollar (solid line) mark (dotted) and franc (dashed) - all relative to 1979.

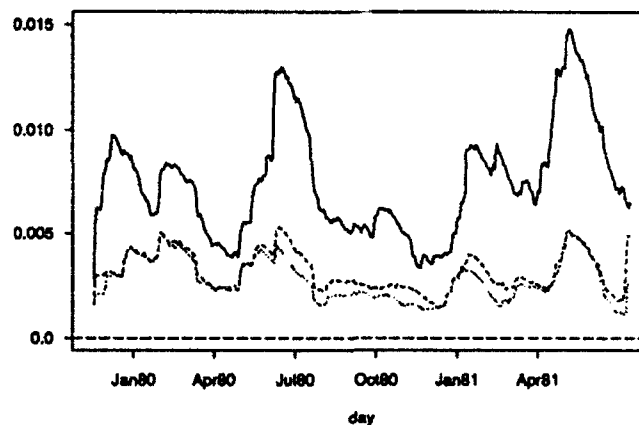


Figure 6: Posterior means $E(\sigma_{ii,t}|I_t)$ for dollar (solid line), mark (dotted) and franc (dashed).

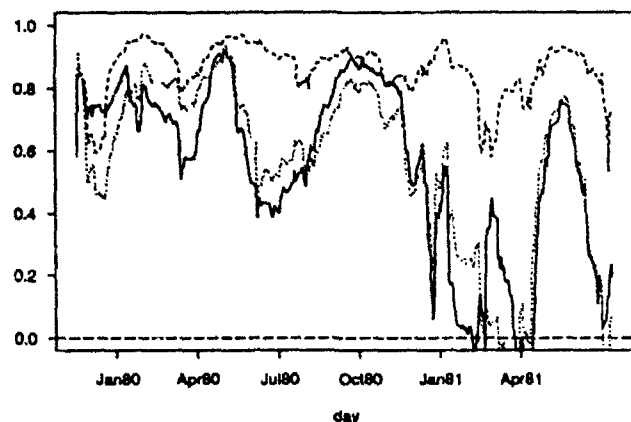


Figure 7: Posterior means for $\text{corr}(D_t, M_t)$ (solid line), $\text{corr}(D_t, F_t)$ (dotted) and $\text{corr}(M_t, F_t)$ (dashed).

posterior means for the ARCH parameters b_t and r_t are shown in Figure 8. The posterior estimates for b_t move within a band from around 0.6 to 0.9, implying that at all times there is strong evidence for an ARCH component in the model.

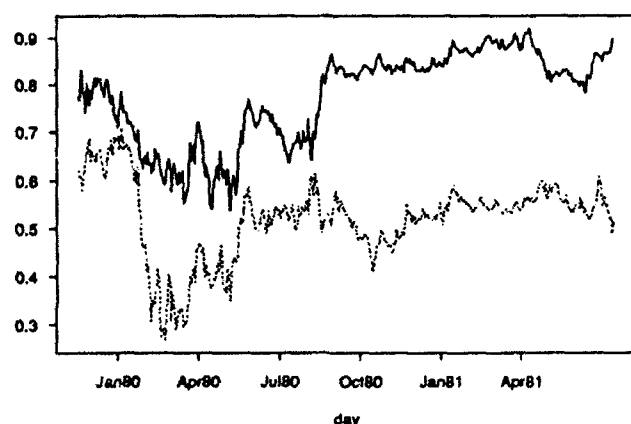


Figure 8: Posterior means for b_t (solid line) and ρ_t (dashed).

References

- Antoniak, C.E. (1974). Mixtures of Dirichlet processes with applications to non-parametric problems. *Annals of Statistics* 2, pp. 1152-1174.
- Berry, D.S. and Christensen, R. (1979). Empirical Bayes' estimation of a binomial parameter via mixtures of Dirichlet processes. *Annals of Statistics* 7, pp. 558-568.
- Diebold, F.X. (1988). *Empirical modelling of exchange rate dynamics*, Lecture Notes in Economics and Mathematical Systems, 303, Springer-Verlag, New York.
- Engle, R. F. (1982). Autoregressive conditional heteroscedasticity with estimates of the variance of United Kingdom inflation. *Econometrica* 50, pp. 987-1008.
- Escobar, M. (1988). *Estimating the means of several normal populations by estimating the distribution of the means*. Thesis, Yale University.
- Escobar, M.D. and West, M. (1990). Bayesian prediction and density estimation. Technical Report 90-A16, ISDS, Duke University.
- Ferguson, T.S. (1973). A Bayesian analysis of some non-parametric problems. *Annals of Statistics* 1, pp. 209-230.
- Friedman, D., Vandersteel, S., and S.E. Fienberg (1985). The Analysis of Cross-Classified Catagorical Data. *Journal of International Economics* 13, pp. 171-186.
- Gerdesmeier, D. (1990). Schweizerische Geld- und Waehrungsgeschichte seit 1973. Discussion Paper 9005, Institut fuer Statistik und Oekonometrie, Universitaet Basel, Basel.
- Hastings, W.K. (1970). Monte Carlo sampling methods using Markov chains and their applications. *Biometrika* 57, pp. 97-109.
- Hsieh, A.D. (1988). The Statistical Properties of Daily foreign exchange rates: 1974-1983. *Journal of International Economics* 24, pp. 129-145.
- Meese, A.R. and Rogoff, K. (1983). Empirical exchange rate models of the seventies. *Journal of International Economics* 14, pp. 3-24.
- Metropolis, N., Rosenbluth, A.W., Rosenbluth, M.N., Teller, A.H., and Teller, E. (1953). Equations of state calculations by fast computing machines. *Journal of Chemical Physics* 21, pp. 1087-1091.
- Migon, H.S. and Harrison, P.J. (1985). An application of non-linear Bayesian forecasting to television advertising. In *Bayesian Statistics 2*, ed. Bernardo, J.M., DeGroot, M.H., Lindley, D.V. and Smith, A.F.M., Valencia.
- Mueller, P. (1991). A Bayesian Vector ARCH Model for Exchange Rate Data. Discussion Paper 9109, Institut fuer Statistik und Oekonometrie, Universitaet Basel, Basel.
- Pole, A. and West, M. (1988). Efficient numerical integration in dynamic models. Technical Report 131, Department of Statistics, University of Warwick.
- Pole, A. and West, M. (1990). Efficient bayesian learning

in nonlinear dynamic models. *Journal of Forecasting* 9 (2), pp. 119-136.

Quintana, J.M. and West, M. (1987). Multivariate time series analysis: New techniques applied to international exchange rate data. *The Statistician* 36, pp. 275-281.

Tierney, L. (1991a). Markov chains for exploring posterior distributions. Technical report 560, University of Minnesota, School of Statistics.

Tierney, L. (1991b). Exploring posterior distributions using Markov chains. in *Computing Science and Statistics: Proceedings on the Interface*, ed. E. Keramidas .

West, M. and Harrison, P.J. (1989). *Bayesian Forecasting and Dynamic Models*. Springer-Verlag, New York.

West, M. (1990a). Bayesian kernel density estimation. Technical Report 90-A02, Duke University, ISDS.

West, M. (1990b). Bayesian computations: sequential and dynamic models. Technical Report 90-A12, Duke University, ISDS.

Mixture Models, Monte Carlo, Bayesian Updating and Dynamic Models

Mike West

Institute of Statistics & Decision Sciences
Duke University, Durham NC 27708, USA

Abstract

This paper reviews the development of discrete mixture distributions as approximations to priors and posteriors in Bayesian analysis, focusing on the development of simulation based techniques in sequential updating and the analysis of dynamic models. Adaptive density estimation techniques enable construction of mixtures of elliptical distributions useful as direct approximations and as importance sampling functions. Illustrations in sequential modelling are discussed.

1. Adaptive mixture modelling

1.1 Importance sampling and mixtures

With the recent mushrooming application of simulation based methods of numerical analysis, Bayesian analyses often involves the approximation of continuous prior and posterior distributions using discrete sets of points and associated weights based on a Monte Carlo approximation. West (1992a) introduced an adaptive importance sampling scheme to develop such discrete approximations, and methods to provide smooth reconstructions, in cases when the prior and likelihood functions separately, or the posterior directly, may be (at least approximately) evaluated up to irrelevant constants. Suppose $p(\theta)$ is the continuous posterior density function for a continuous parameter vector θ . An approximating density $g(\theta)$ is used as an importance sampling function (Geweke, 1989) as follows. Let $\Theta = \{\theta_j, j = 1, \dots, n\}$ be a random sample from $g(\theta)$, and define weights $\Omega = \{w_j, j = 1, \dots, n\}$ by $w_j = p(\theta_j)/(kg(\theta_j))$, for each j , where $k = \sum_{j=1}^n p(\theta_j)/g(\theta_j)$. The weights are evaluated via $w_j \propto p(\theta_j)/g(\theta_j)$, and then normalised to unit sum. Then inference under $p(\theta)$ is approximated using the discrete distribution having masses w_j at θ_j , for each $j = 1, \dots, n$. The conditions on $g(\theta)$ required to achieve reasonable approximations typically reduce to requiring g have the same support as p and that the tails of g be heavier than those of p , so that variations on multivariate T distributions have become popular (Geweke, 1989). In West (1992a) mixtures of T distributions are proposed, one additional reason being that mixtures have the flexibility to represent the possibly quite complex and varied forms of (posterior) densities. This is done using kernel

density estimation techniques. With an importance sampling function $g(\theta)$ close to the true density $p(\theta)$, kernel density estimation (or other smoothing techniques) provides continuous estimates of joint and marginal densities. In simple, univariate random sampling problems, kernel type density estimates have direct Bayesian interpretation as approximate predictive distributions in models based on mixtures of Dirichlet processes (West, 1990); multivariate analogues are similarly derivable (Erkanli, Müller and West, 1992). West (1992a) uses weighted variations on multivariate kernel estimates as importance sampling functions and, with some modification, to more directly estimate marginal densities of $p(\theta)$. The basic idea is as follows.

Given a chosen importance sampling density $g_0(\theta)$, the sample, of size n , Θ and associated weights Ω , the exact density $p(\theta)$ may be approximated by a weighted kernel estimate of the form

$$g_1(\theta) = \sum_{j=1}^n w_j d(\theta|\theta_j, Vh^2) \quad (1)$$

where $d(\theta|\mathbf{m}, \mathbf{M})$ denotes a p -variate, elliptically symmetric density function (determining the 'kernel'), with mode \mathbf{m} and scale matrix \mathbf{M} , \mathbf{V} is an estimate of the variance matrix of $p(\theta)$, (usually the Monte Carlo estimate based on Θ and Ω), and h is a 'window-width' smoothing parameter, depending on the Monte Carlo sample size n . A key example has $d(\theta|\mathbf{m}, \mathbf{M})$ as the density of a multivariate T distribution with some $\alpha > 0$ degrees of freedom, whence the density $g_1(\theta)$ is a discrete mixture of n component T distributions. Conventional density estimation techniques (Silverman, 1986) choose the window width h as a slowly decreasing function of n , so that the kernel components are naturally more concentrated about the locations θ_j for larger sample sizes, and the approach of West (1990) indicates how currently popular rules for choosing h may be interpreted within a Bayesian and, more importantly, model based framework. If h does decay to zero at the right rate as n increases, then $g_1(\theta)$ approaches $p(\theta)$ as n increases. For moderate sample sizes, $g_1(\theta)$ will tend to be overdispersed relative to $p(\theta)$ if \mathbf{V} is, as is usual, the Monte

Carlo estimate of $V[\theta]$ under $p(\theta)$. This feature proves useful in developing kernel forms as importance density functions, though can be counter-balanced using 'shrinkage' of kernel locations to provide more direct approximation of the true density and its margins (West, 1992a).

Adaptive methods of posterior approximation based on the generalised kernel technique are now discussed. Note first, however, that there are obvious ways in which they may require modification in specific applications. A key issue concerns the possible patterns of local dependence exhibited by $p(\theta)$, and the use in (1) of a global estimate of covariance. In many problems, this will be quite adequate, but it is not uncommon to encounter problems in which different regions of parameter space are associated with rather different patterns of dependence, so that a local estimate of covariance structure, with V in (1) varying with locale j and more heavily depending on θ_i values near θ_j , is required. Other modifications involve the more formal modelling foundation for kernel type techniques that derive from the approach to density estimation using Dirichlet process mixtures of normal distributions (West, 1990; Escobar and West, 1992). Current developments of the dynamic modelling applications using these ideas are mooted in Müller (1993). Finally, some of the modifications mentioned in West (1992a), especially a neat data augmentation trick used to effect dimensionality reduction and significantly reduce computations in multiparameter models, may be directly applied in sequential problems.

1.2 Adaptive importance sampling

Adaptive importance sampling describes any process by which the importance sampling distribution is sequentially revised based on information derived from successive Monte Carlo samples. Let $g_0(\theta)$ be an initially chosen importance sampling function. For a sample size n , this leads to points $\Theta_0 = \{\theta_{0,j}, j = 1, \dots, n_0\}$, and weights $\Omega_0 = \{w_{0,j}, j = 1, \dots, n_0\}$, and the summary

$$\Gamma_0 = \{g_0, n_0, \Theta_0, \Omega_0\}.$$

Adaptive importance sampling suggests taking n_0 rather small, say several hundreds, and, based on the Monte Carlo outcome Γ_0 , revising $g_0(\theta)$ to a 'better guess'. It is natural to use a mixture such as (1) as a second step importance sampling function, and the following adaptive routine arises:

- (1) Choose an initial importance sampling distribution with density $g_0(\theta)$, draw a fairly small sample n_0 and compute weights, deducing the summary $\Gamma_0 = \{g_0, n_0, \Theta_0, \Omega_0\}$. Compute the Monte Carlo estimates $\bar{\theta}_0$ and V_0 of the mean and variance of $p(\theta)$.

- (2) Construct a revised importance function $g_1(\theta)$ using (1) with sample size n_0 , points $\theta_{0,j}$, weights $w_{0,j}$, and variance matrix V_0 .
- (3) Draw a larger sample of size n_1 from $g_1(\theta)$, and replace Γ_0 with $\Gamma_1 = \{g_1, n_1, \Theta_1, \Omega_1\}$.
- (4) Either stop, and base inferences on Γ_1 , or proceed, if desired, to a further revised version $g_2(\theta)$, constructed similarly. Continue as desired.

Even though the initial $g_0(\theta)$ may poorly represent $p(\theta)$, successive refinement through smaller samples can, and usually does, mean that, after one or two revisions, the resulting kernel estimate is a much better approximation to $p(\theta)$. Hence, once the process of refinement is terminated, a much smaller sample size is necessary for the desired accuracy of approximation. Often just one refinement is sufficient to adjust a very crude approximation, $g_0(\theta)$, say a single multivariate T density, to a mixture $g_1(\theta)$ of, say, 500 T densities, that much more closely represents the true $p(\theta)$. In approximating moments and probabilities, a Monte Carlo sample of two or three thousand draws from $g_1(\theta)$ may do as well as, or better than, a sample of two or three times that from the original $g_0(\theta)$. Useful diagnostic information is generated in this process at each stage, such as the configuration of points in each dimension of the parameter space, and the distributions of weights. This can be used to guide successive choice of sample sizes, and possible interventions to adjust successive kernel smoothing parameter values, and also to assess whether further refinement is likely to be additionally effective. Several illuminating examples appear in West (1992a).

1.3 Approximating mixtures by mixtures

Suppose the above adaptive strategy has $n_0 = 500$, so that $g_1(\theta)$ is a mixture of 500 p -dimensional T distributions, and that the revision process stops here, $g_1(\theta)$ being adopted as the 'final' importance sampling density to be used for Monte Carlo inference. It is straightforward to sample $\Theta_1 = \{\theta_{1,i}, i = 1, \dots, n_1\}$ from $g_1(\theta)$ - the computational benefit of the components sharing a common scale matrix V_0 is apparent here. It is also straightforward to then evaluate the weights $\Omega_1 = \{w_{1,i}, i = 1, \dots, n_1\}$, though the denominator of $w_{1,i}$ requires evaluation of the mixture $g_1(\theta_{1,i})$. The computational burden here clearly increases if further refinement of importance functions with rather larger sample sizes, is desired. One way of reducing such computations is to note that, quite typically, approximating $p(\theta)$ using a mixtures of several thousand T, or other, distributions is really overkill; even very irregular densities can be adequately matched using mixtures having far fewer components. The Monte Carlo kernel density construction, in particular, typically leads to a huge re-

dundancy, with many points $\theta_{0,j}$ closely grouped and contributing essentially similar components to the overall mixture density. In the context of raw density estimation, West (1990) discusses the reduction of kernel estimates to mixtures of much smaller numbers of components, often lower than 10% of the original sample size number, using a particular form of clustering. The discussion in West and Harrison (1989, Section 12.3), on issues and techniques involved in approximating mixtures generally, is also relevant.

A very basic method of 'clustering' mixture components, combining ideas from each of these two references, is used in West (1992a). At the simplest, it involves reducing the number of components by replacing 'nearest neighbouring' components with some form of average. The examples below, and those in West (1992a), involve reducing mixtures of n in several hundreds or thousands, to around 10% (though sometimes rather less) of the initial value, and perform this reduction using the following simple clustering routine (see West, 1992a, for further details).

- (1) Set $r = n$, and, starting with the $r = n$ component mixture (1), choose $k < n$ as the number of components for the final, reduced mixture.
- (2) Sort the r values of θ_j in Θ in order of increasing values of weights w_j in Ω ; thus θ_1 corresponds to the component with smallest weight.
- (3) Find the index i , ($i = 1, \dots, r$), such that θ_i is the nearest neighbour of θ_1 , and reduce the sets Θ and Ω to sets of size $r - 1$ by removing components 1 and i , and inserting 'average' values $\theta_* = (w_1\theta_1 + w_i\theta_i)/(w_1 + w_i)$ and $w_* = w_1 + w_i$. Set $r = r - 1$.
- (4) Proceed to (2), and repeat the procedure, stopping here only when $r = k$.
- (5) The resulting mixture has the form (1) with n reduced to k , the locations based on the final k averaged values, with associated combined weights, the same scale matrix V but new, and larger, window-width h based on the current, reduced 'sample size' r rather than n .

The reduction process can be monitored, by evaluating and plotting margins of the mixture over fairly crude grids, at stages in the reduction process. If in reducing from, say, n to 80% n components, these densities do not appear change much, then further reduction may proceed, and so forth in later stages of reduction. This is one possible route to approximating mixtures of large numbers of components with mixtures of far fewer. There are other possibilities, using alternative clustering algorithms, such as those and with foundation in Bayesian density estimation models (West, 1990), as illustrated by Müller (1993).

2. Sequential updating and dynamic models

2.1 Introduction

A central concept in Bayesian analysis is that of updating a prior to posterior distribution for a random quantity or parameter vector θ based on received data summarised through a likelihood function for the parameter. With discrete approximations, this involves mapping a prior set of points and weights to a posterior set, possibly with some form of smoothing involved at both prior and posterior stages. These issues, and others, are sharply evident in sequential modelling of time series using dynamic models (West and Harrison, 1989), where the progressive revision of posterior and predictive distributions requires calculations that are typically impossible to perform exactly. Modellers have developed a variety of approaches to analytic and numerical approximation of such distributions, including direct quadrature methods (Kitagawa, 1987) and more efficient adaptive methods (Pole and West, 1990). The current paper extends the adaptive importance sampling techniques just reviewed to the dynamic modelling context to overcome the problems associated with quadrature techniques: notably, the need for 'grids' of evaluation points in parameter spaces to be changed as time progresses and as data indicate support for different regions in parameter space, the severe computational demands in problems with more than very few parameters, and the difficulties in reconstructing smooth posterior distributions based on approximate evaluation at only very few points in what may be several dimensions. These issues currently limit the development of quadrature techniques quite generally, outside the dynamic modelling framework, and have led investigators in other fields to turn to simulation based approaches of one form or another. Building on these approaches, the development in West (1992a) of adaptive importance sampling functions based on mixtures of standard distributions, and smooth reconstruction of posteriors using generalised kernel density estimates, may be naturally extended to the sequential dynamic modelling context.

2.2 Dynamic models

Interest lies in analysis of a dynamic model for a time series of (possibly multivariate) observations Y_t , ($t = 1, 2, \dots$), described as follows (West and Harrison, 1989, Section 13.5). At each time t , Y_t has a known sampling distribution with density (or mass function) $p_o(Y_t|\theta_t)$; here θ_t is a p -vector of parameters, the state vector, and, conditional on θ_t , Y_t is assumed independent of Y_s and θ_s for all past and future values of s . The state vector is assumed to evolve in time according to a known, Markovian process described by an evolution density $p_e(\theta_t|\theta_{t-1})$; given θ_{t-1} , θ_t is conditionally inde-

pendent of past values θ_s for $s < t - 1$. Summarising the model, for each $t = 1, 2, \dots$, the defining equations are

$$\text{Observation model: } (Y_t | \theta_t) \sim p_o(Y_t | \theta_t) \quad (2)$$

$$\text{Evolution model: } (\theta_t | \theta_{t-1}) \sim p_e(\theta_t | \theta_{t-1}) \quad (3)$$

The sampling, or observation, density is suffixed 'o', and the evolution density is suffixed 'e', in order to identify them in the discussion below where 'p' is used generically to denote densities. Note that these densities may depend also on t and on other available historical information, though the notation ignores this for clarity. At each time t , available information, including the observed values of Y_t, Y_{t-1} , etc, is denoted D_t . Currently, at time $t - 1$ prior to evolution through (3), historical information is summarised through (an approximation to) the posterior $p(\theta_{t-1} | D_{t-1})$. The primary computational problems addressed are:

- (a) **Evolution step:** compute the current prior for θ_t , defined via

$$p(\theta_t | D_{t-1}) = \int p_e(\theta_t | \theta_{t-1}) p(\theta_{t-1} | D_{t-1}) d\theta_{t-1}; \quad (4)$$

- (b) **Updating step:** on observing Y_t , compute the current posterior

$$p(\theta_t | D_t) \propto p(\theta_t | D_{t-1}) p_o(Y_t | \theta_t). \quad (5)$$

Subsidiary calculations include forecasting ahead (discussed below) and filtering problems for retrospection (not considered in this paper). It is important to note that analysis proceeds sequentially over time, and it is explicitly recognised that the summary posterior of historical information, namely $p(\theta_{t-1} | D_{t-1})$, is the only information relevant to the past that is currently (at time $t - 1$) available for further analysis, and this must be borne in mind when developing numerical (or any other) approximations to dynamic model analyses. In addition, the process is to be repeated as time progresses, so the form of any approximation to the posterior after the updating step must be such that, moving from time t to $t + 1$ when this posterior becomes the input to the evolution/updates analysis for θ_{t+1} , the required calculations can proceed similarly. In developing Monte Carlo based approximations (or any other numerical approach) the objective is therefore to provide structured information about $(\theta_t | D_t)$, after the updating step, that has the same qualitative form as that input for $(\theta_{t-1} | D_{t-1})$. With reference to development above, a Monte Carlo/importance sampling approximation to any distribution reduces to summary components given by the four elements Γ , so that, with such an approach to

analysis, the entire evolution/updates process at each time can be viewed as one of appropriately updating such components, with subsidiary computations for forecasting, etc. This directly parallels the usual, analytic process in linear, normal models, (West and Harrison, 1989), and the numerical process using quadrature of Pole and West (1990).

Suppose that $p(\theta_{t-1} | D_{t-1})$ has been previously approximated via $\Gamma_{t-1} = \{g_{t-1}, n_{t-1}, \Theta_{t-1}, \Omega_{t-1}\}$, where $g_{t-1}(\theta_{t-1})$ is a (final) importance density used for inference about $(\theta_{t-1} | D_{t-1})$, n_{t-1} is the (final) Monte Carlo sample size in that inference, $\Theta_{t-1} = \{\theta_{t-1,i}, i = 1, \dots, n_{t-1}\}$ is the sample from $g_{t-1}(\theta_{t-1})$, and $\Omega_{t-1} = \{w_{t-1,i}, i = 1, \dots, n_{t-1}\}$ the associated weights. The objective is to perform the evolution/updates and forecasting computations and finally summarise $p(\theta_t | D_t)$ in terms of $\Gamma_t = \{g_t, n_t, \Theta_t, \Omega_t\}$, and this may be done as follows.

2.3 Computations: evolution step

The following facts are of use in computations for the evolution step.

- (a) Various features of the prior $p(\theta_t | D_{t-1})$ of interest can be computed directly using the Monte Carlo structure Γ_{t-1} . The prior mean, for example, is computable as

$$E[\theta_t | D_{t-1}] \approx \sum_{i=1}^{n_{t-1}} w_{t-1,i} E_e[\theta_t | \theta_{t-1,i}],$$

where E_e stand for expectation under the evolution distribution p_e , if this expectation is available in closed form.

- (b) Similarly, the prior density function can be evaluated by Monte Carlo integration at any required point, viz

$$p(\theta_t | D_{t-1}) \approx \sum_{i=1}^{n_{t-1}} w_{t-1,i} p_e(\theta_t | \theta_{t-1,i}). \quad (6)$$

- (c) An initial Monte Carlo sample of size n_{t-1} may be drawn from the prior by generating one value of θ_t from $p_e(\theta_t | \theta_{t-1,i})$, for each $i = 1, \dots, n_{t-1}$. The resulting sample points, denoted Θ_t^* , provide, at the very least, starting values for the evaluation of the prior as in (b) above, for display purposes, and, importantly, regions of θ_t parameter space to concentrate on in beginning the updating step, below.
- (d) This prior sample Θ_t^* may be used with weights Ω_{t-1} to construct a generalised kernel density estimate of the prior. In many models, this is of little interest since the prior may be evaluated as

in (b) above (unless the evolution density is extremely complex and difficult to work with.) Consider, however, a model in which the evolution equation is degenerate, $\theta_t = G_t(\theta_{t-1})$ with probability one, for some known, possibly non-linear and time-dependent function G_t . This covers many interesting examples, such as that appearing in Section 4.4 below. Then prior evaluation as in (b) is vacuous, and so, since values of the prior are required as input to Bayes' theorem (5) in the updating step, some form of interpolation/smoothing is needed.

- (e) Note, at this point, that subsidiary computations for forecasting ahead can be performed along these lines. Forecasting $(Y_t|D_{t-1})$, for example, requires computing $\int p_o(Y_t|\theta_t)p(\theta_t|D_{t-1})d\theta_t$. The observation density typically has a standard form, so that Monte Carlo computations can be performed to approximate forecast moments and probabilities. For example, the forecast mean is evaluated as

$$E[Y_t|D_{t-1}] \approx \sum_{\theta_t \in \Theta_t^*} w_{t-1,i} E_o[Y_t|\theta_t],$$

where E_o stand for expectation under the observation distribution p_o , assuming this expectation is available in closed form. Forecast probabilities are similarly derived. By simulating from the observation density for each value of $\theta_t \in \Theta_t^*$, and using the associated weights Ω_{t-1} , regions of interest in Y_t space can be identified, and the predictive density/mass function evaluated there. Similar comments apply to forecasting more than one step ahead; the process of simulating from the evolution density is repeated into the future, generating sample of future parameter vectors, and proceeding to inference about the future values of the series. This too is illustrated below.

It is important to note the generality of the above strategy for computations. At no stage is it necessary, or interesting, to worry about functional forms of evolution equations, or to cater for many special cases, as alternative approaches, such as using quadrature in Pole and West (1990), must. This simplifies programming the analysis; all that is needed is a collection of general routines for evaluating and sampling from the evolution and observation densities, and, if required, generating kernel estimates.

2.4 Computations: updating step

Following evolution, the prior $p(\theta_t|D_{t-1})$ to input to Bayes' theorem (5) for updating is available in approximate form either via evaluation in (6) at any desired points, or in terms of a generalised kernel form as described in (d) of the previous section. Updating may

then proceed using adaptive Monte Carlo density estimation of Section 2. This begins with an initial 'guess' at the posterior $p(\theta_t|D_{t-1})$, denoted $g_{t,0}(\theta_t)$, to be used as an initial importance sampling function. Choice of this density depends on context, though may be guided by general ideas described in the examples below. This provides the starting point for application of the adaptive strategy of section 1. On completion, this results in a final summary of the posterior given by the quadruple $\Gamma_t = \{g_t, n_t, \Theta_t, \Omega_t\}$, where $g_t(\theta_t)$ is the final importance sampling density for the posterior $p(\theta_t|D_t)$, n_t is the final Monte Carlo sample size, and Θ_t is the set of n_t points in parameter space at which the Monte Carlo weights in Ω_t are evaluated.

3. EXAMPLES

Example 1

The first example concerns a familiar and simple model, chosen so that exact calculations can be performed to provide comparison with approximation based on the above strategy. The model is a normal, linear, first-order polynomial model (West and Harrison, 1989, Chapter 2), in which $p = 1$, so that $\theta_t = \theta_t$ is scalar, the observation density is normal with mean θ_t and unit variance, $(Y_t|\theta_t) \sim N[\theta_t, 1]$, and the evolution density is normal, $(\theta_t|\theta_{t-1}) \sim N[\theta_{t-1}, 1]$. If it is assumed, initially, that $(\theta_1|D_0)$ is normal, then the standard analysis applies to give easy calculation of prior, posterior and forecast distributions for any time t , such distributions being, of course, normal. The example here uses a series of length 100 generated from this model with $\theta_0 = 0$. The numerical analysis is structured as follows: (i) The initial prior is standard T with 9 degrees of freedom; (ii) in each updating step, computing the prior densities $p(\theta_t|D_{t-1})$ for evaluation of the (unnormalised) posteriors is done via equation (6); (iii) updating uses adaptive Monte Carlo density estimation, the sample size of all final approximations set at 1500 – the first approximation $g_{t,0}(\theta_t)$ at each stage generated as follows. The summary Γ_{t-1} for $(\theta_{t-1}|D_{t-1})$ (with sample size $n_{t-1} = 1500$) is used to generate an initial sample $\Theta_t^* = \{\theta_{t,i}^*, i = 1, \dots, n_{t-1}\}$ for θ_t , the estimate of mean and variance given by $a_t = \sum_{i=1}^{n_{t-1}} w_{t-1,i} \theta_{t,i}^*$, and $R_t = \sum_{i=1}^{n_{t-1}} w_{t-1,i} (\theta_{t,i}^* - a_t)^2$, respectively. Clustering reduces this $n_{t,0} = 100$ components $\Theta_{t,0} = \{\theta_{t,0,j}, j = 1, \dots, 100\}$, with reduced weights $\Omega_{t,0} = \{w_{t,0,j}, j = 1, \dots, 100\}$. Then $g_{t,0}(\theta_t)$ is taken as the mixture (1) using these reduced weights and points, the variance $V = 16R_t$, window-width h chosen as in Silverman (1986) but using a T_9 kernel. The factor 16 inflates the spread to provide a conservative initial guess. The adaptive strategy now proceeds with $n_{t,0} = 250$, $n_{t,1} = 350$ and, fi-

nally, $n_t = n_{t,2} = 1500$. After sampling and computing moments in the first two stages, the components, of sizes 250 and 350 respectively, are reduced to 100 by clustering before sampling for the next iteration.

Inferences at each stage are based on the final, full Monte Carlo summaries of size 1500. Some posterior densities at times $t = 10, 20, \dots, 90$ appear in Figure 1, with the current, actual value of θ_t appearing on the axis as 'X', the corresponding datum Y_t as 'Y'. In fact each frame plots two densities, the second being the exact, normal posterior based on the usual normal theory analysis. In none of these plots, and indeed over all time $t = 1, \dots, 100$, is there a meaningful difference between the exact and numerical approximation.

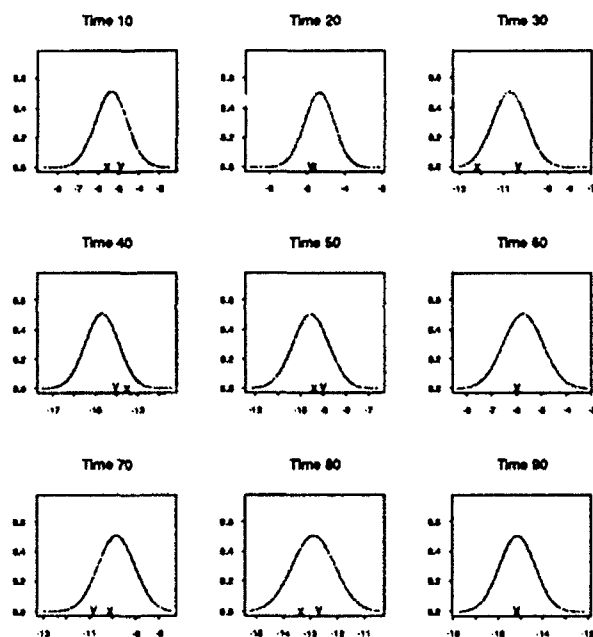


Figure 1. Posterior densities for normal process

Example 2

A similar example, though not normal, is analysed in West (1992b) – the normality of evolution and observation errors replaced with Student T distributions. Such models have been popular in the literature (Kitagawa, 1987; Pole and West, 1990, and references therein; West, 1981). This permits larger 'jumps' in the level θ_t of the time series Y_t , and also accommodates outlying observations. West (1992b) demonstrates the efficacy of the adaptive mixture techniques in analysing such a model. Unlike the normal case above, the posteriors for level parameters can become heavily skewed and even bimodal, evidencing occurrences of very extreme obser-

vations that induce 'conflict' between priors and likelihood functions. The facilities for outlier rejection and adaptation to level changes in heavy-tailed models are illustrated in West (1992b) using the numerical methods described here.

Example 3

A simplified version of a model of Andrade Netto *et al* (1978), discussed in Kitagawa (1978), illustrates uniparameter analyses further, with some interesting features. The model is

$$(Y_t|\theta_t) \sim N[0.05\theta_t^2, 10],$$

$$(\theta_t|\theta_{t-1}) \sim N[0.5\theta_{t-1} + 25\theta_{t-1}/(1 + \theta_{t-1}^2), 1].$$

The evolution structure here is bifurcating; the conditional mean $E_\theta[\theta_t|\theta_{t-1}]$ near $\theta_{t-1} = 0$ is approximately linear with gradient 25.5, so that small values of θ_{t-1} tend to evolve to very much larger values of θ_t ; as a result, very regular posteriors $p(\theta_{t-1}|D_{t-1})$, unimodal near zero, evolve into bimodal priors $p(\theta_t|D_{t-1})$, the mass being pushed away from the origin through the deterministic component of the evolution equation. The observation equation leads to further irregularities. For negative observed values of Y_t , the likelihood function, proportional to $p_\theta(Y_t|\theta_t)$ is unimodal at zero, and symmetric about its mode; for positive observations, however, the likelihood has an antinode at zero, and is symmetric about zero with modes at $\pm\sqrt{20Y_t}$.

The analysis for just three observations generated from this model is summarised here. The process is simulated from $\theta_0 = 0$, the three simulated θ_t values, and the corresponding simulated Y_t values, noted on the lower axes of the frames of Figure 2. Analysis is as follows. (i) The initial prior for $(\theta_1|D_0)$ is Student T on 9 degrees of freedom, with mode at 0 and scale factor 100; thus $\theta_1/10$ is standard T_9 . (ii) Values of the prior densities $p(\theta_t|D_{t-1})$, required for evaluating the (unnormalised) posteriors in Bayes' theorem (5), are calculated using the exact evolution density in the Monte Carlo expression (6). (iii) Updating uses adaptive Monte Carlo density estimation, the sample size of all final approximations set at 2000. The first approximation $g_{1,0}(\theta_t)$ at each stage being generated as follows. First, Γ_{t-1} (with sample size $n_{t-1} = 2000$) is used to generate an initial sample $\Theta_t^* = \{\theta_{t,i}^*, i = 1, \dots, n_{t-1}\}$ for θ_t . Clustering then applies to reduce this from 2000 to $n_{t,0} = 100$ components $\Theta_{t,0} = \{\theta_{t,0,j}, j = 1, \dots, 100\}$, with associated reduced weights $\Omega_{t,0} = \{w_{t,0,j}, j = 1, \dots, 100\}$. Then $g_{1,0}(\theta_t)$ is taken as the mixture (1) using these reduced weights and points, the variance $V = R_t$, and the kernel is again T_9 . The adaptive strategy now proceeds using $n_{t,0} = 250$, $n_{t,1} = 500$ and, finally, $n_t = n_{t,2} = 2000$. Finally, the components, of sizes 250 and 500 respectively, are reduced to 100 before sampling for the next iteration.

For $t = 1, 2$ and 3 , the evaluated prior $p(\theta_t|D_{t-1})$, and the likelihood function proportional to $p_o(Y_t|\theta_t)$ (the latter standardised to unity at the maximum), appear in

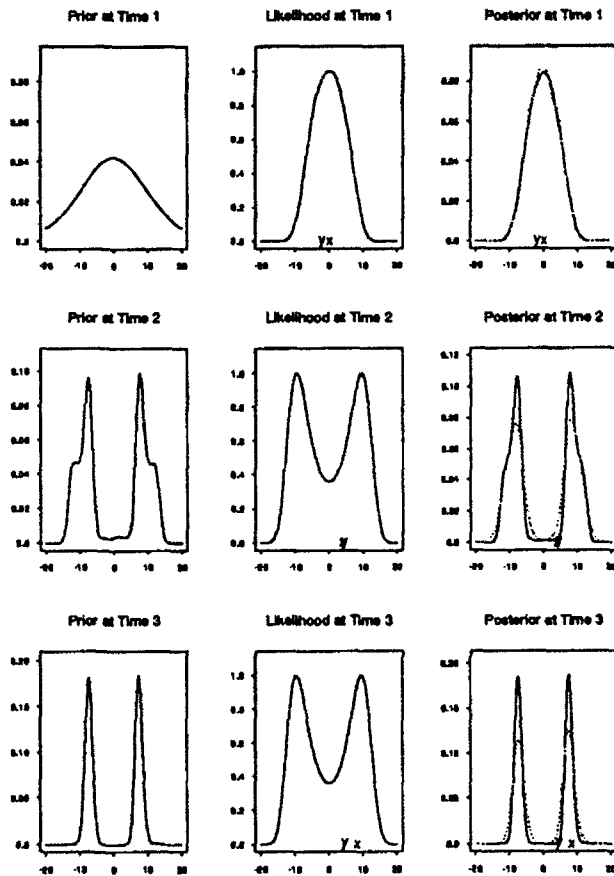


Figure 2. Inference in bifurcating model

the first two frames of each row of graphs in Figure 2. The third frame in each row displays, as a broken line, the final density $g_t(\theta_t)$, a mixture of 100 T_9 densities. As a full line in the final frames, the 'exact' posterior based on the displayed prior and likelihood function is graphed. This is computed as the approximately normalised product of the prior and likelihood, with evaluations made over a fine grid of 300 points across the displayed interval of ± 20 . In the cases of extreme bimodality, the importance density is naturally less peaked at the modes, though probabilities computed under the two densities are substantially similar. Continuing analysis to $t = 4, 5, \dots$, this pattern of behaviour is reproduced, with the two densities in the third frame essentially coinciding apart from cases of extreme bimodality when minor differences appear around the modes. Quan-

tified inferences are based on the actual Monte Carlo approximations Γ_t , rather than by using the importance densities as approximations directly, so that the underestimation of density heights near peaks is irrelevant. As importance sampling densities for the displayed 'exact' posteriors, the displayed forms are excellent. Very similar results are achieved by repeating the analysis with larger sample sizes in the Monte Carlo. With the adaptive kernel density estimation used here, samples of 2000 at the final stage are typically quite adequate.

As final comment, it may be shown that, in fact, the prior and posterior distributions in the example are theoretically guaranteed to be symmetric about zero, for all time t , when the initial prior $p(\theta_1|D_0)$ is itself symmetric about zero (as is the case in the above analysis.) None of the estimated densities in Figure 2 departs significantly from symmetry about zero, although, due to the randomness inherent in Monte Carlo approaches, small deviations away from symmetry are apparent. It should be noted that knowledge of such a feature, or any other theoretically based information about form, may be incorporated into the analysis if required. The symmetry about zero here could easily be incorporated by taking absolute values of all sampled θ_t , and then reflecting resulting samples, and smoothed densities, about the origin.

Example 4

A final example revisits a class of models used in assessing the effect of television advertising on the 'awareness' of consumer populations, introduced in Migon and Harrison (1985), and discussed in West and Harrison (1989, Section 14.4). The particular model chosen for illustration here is one in which prior evaluations using (6) are impossible since the evolution distribution is degenerate. This example thus provides (a) a multi-parameter model, with $p = 3$; (b) an opportunity to explore numerical analysis in this familiar model class, and establish a basis for possible future comparison with previous analytic approximations; and, (c) a case in which direct generalised kernel estimation is used to sequentially reconstruct posterior distributions from Monte Carlo analyses.

Much detail on the models and context of the advertising problem appears in the above references, to which the reader may refer, although the data analysis here is somewhat different to previous analyses. Raw 'advertising awareness' data, in the form of binomial counts, are collected over a period of 56 weeks, together with an associated regressor variable 'TVR' measuring weekly advertising expenditure. Indexing weeks by k , ($k = 1, \dots, 56$), let Z_k denote recorded counts in week k , and $X_k > 0$ the 'TVR'. The model assumes Z_k to be

conditionally independent with binomial probabilities π_k defined (using the notation of West and Harrison, 1989, Section 14.4) via

$$\begin{aligned}\pi_k &= 0.1 + E_k, \\ E_k &= 0.7 - (0.7 - \rho E_{k-1}) \exp(-\kappa X_k),\end{aligned}\quad (7)$$

for $k = 1, \dots, 56$. Quantities ρ and κ , and the initial effect E_0 , are initially uncertain. The data are the first 56 weeks of series in Table 14.1, Section 14.4 of West and Harrison (1989). Analysis here groups the raw weekly data into seven, 8-week periods, defining data Y_t via

$$Y_t = \{Z_{8t-7}, Z_{8t-6}, \dots, Z_{8t}\}, \quad (t = 1, \dots, 7).$$

For each t , the three uncertain parameters are ρ , κ and E_{8t-8} , the effect E_{8t-8} being that in the final week of the previous 8-week period. In the first 8-week period, for example, $t = 1$ so that $E_{8t-8} = E_0$, the current effect just prior to starting the advertising campaign. The sampling density in period t is the product of 8 component binomials, $\prod_{k=8t-7}^{8t} p(Z_k | \pi_k)$, in which the π_k can be evaluated as functions of ρ , κ , and E_{8t-8} by repeat application of equations (7). In evolution to times $t = 2, \dots, 7$, ρ and κ remain unchanged, and the third parameter E_{8t-8} is a deterministic, non-linear function of that previously, E_{8t-16} ; given E_{8t-16} , evolution calculations again just involve repeat application of the second equation in (7) to evaluate E_{8t-8} . Note that the parameters are restricted by inequalities $1 > \rho > 0$, $\kappa > 0$ and $0.7 > E_{8t-8} > 0$. Since importance sampling densities are to be constructed as mixtures of tri-variate T distributions, the restricted parameters ρ , κ and E_{8t-8} are transformed to real-valued versions $\theta_t = (\theta_1, \theta_2, \theta_{t,3})'$ where $\theta_1 = \log(\rho/(1-\rho))$, $\theta_2 = \log(\kappa)$ and $\theta_{t,3} = \log(E_{8t-8}/(0.7 - E_{8t-8}))$. The analysis summarised here (validated by repeat analysis using different, and larger, Monte-Carlo sample sizes at each stage) has the following features, again essentially similar in structure to previous examples.

- (i) The initial prior for $\theta_1 = (\rho, \kappa, E_0)'$ given D_0 is trivariate T with 9 degrees of freedom; the prior mean is $(2.0, -3.5, -0.75)'$, and the prior scale factors are $(0.20, 0.25, 0.25)'$, so $V[\theta_1 | D_0] = 0.20(9/7)$ and $V[\theta_2 | D_0] = V[\theta_3 | D_0] = 0.25(9/7)$. The data set used here is taken from Section 14.4 of West and Harrison (1989), and this prior specification is consistent with that of the analysis there. The prior has no non-zero correlations.
- (ii) The Monte Carlo summary Γ_{t-1} of $p(\theta_{t-1} | D_{t-1})$ evolves by just mapping the third elements of vectors in $\Theta_{t-1} = \{\theta_{t-1,i}, i = 1, \dots, n_{t-1}\}$. Thus Γ_{t-1} evolves to the prior summary Γ_t^* via just this,

deterministic transformation. Values of the prior $p(\theta_t | D_{t-1})$ required for evaluating the (unnormalised) posteriors in Bayes' theorem (5), are calculated from a kernel density reconstructed version, based on this evolved Monte Carlo approximation Γ_t^* and using trivariate T_9 kernels.

- (iii) Updating uses adaptive Monte Carlo density estimation, the sample size of all final approximations set at 4000. The first approximation $g_{t,0}(\theta_t)$ at each stage is the weighted kernel estimate based on Monte Carlo summary Γ_t^* but with the variance matrix scaling the kernels inflated by a factor of 4; this has the same location characteristics as the reconstructed prior $p(\theta_t | D_{t-1})$, but greater spread. The process of adaptive refinement proceeds through two stages, as in the examples above, with sample sizes successively increased from 1000, 1500 and then the final 4000. Between stages, the clustering technique of Section 2.4 is applied to reduce the number of components (from 1000 and 1500 respectively) to 500.

At the end of analysis at each stage $t = 1, \dots, 7$, the final summary Γ_t of $p(\theta_t | D_t)$ is evolved to Γ_{t+1}^* and further evolution over the $(t+1)^{\text{st}}$ 8-week period is performed to produce step-ahead forecasts of the probabilities π_k in (7); thus, at time t , direct transformation of Γ_t leads to Monte Carlo predictive distributions for π_k for each $k = 8t-7, 8t-6, \dots, 8t$. Figure 3(b) displays the TVR inputs X_k versus time k . In Figure 3(a), the raw data proportions $Z_k/66$ are indicated at each time k , joined up over time with a full line. The step-ahead forecast distributions for π_k , generated at the end of each 8-week period (marked by vertical dotted lines) for the coming 8 weeks, are indicated in terms of medians (the small circles) and vertical lines representing 90% equal-tails intervals. Note that data variation about the π_k is binomial; these intervals are for the π_k directly, so forecast intervals for the Z_k would be similarly located but wider due to the additional binomial variation. At these levels of π_k , around 0.3-0.4, binomial standard deviations are around 0.06. The analysis compares well with similar displays (albeit from a rather different model and analysis) in West and Harrison (1989, Section 14.4).

With the time independence of the parameters ρ and κ , the data could alternatively be analysed as one, rather than sequentially. Here, to illustrate the techniques in sequential modelling, it has been assumed that summary inferences are required after each 8-week period, and that this summary at each stage is all that is carried forward to the next; previous data, and therefore any opportunity to perform revised analyses using all data so far, is assumed lost at the end of each stage.

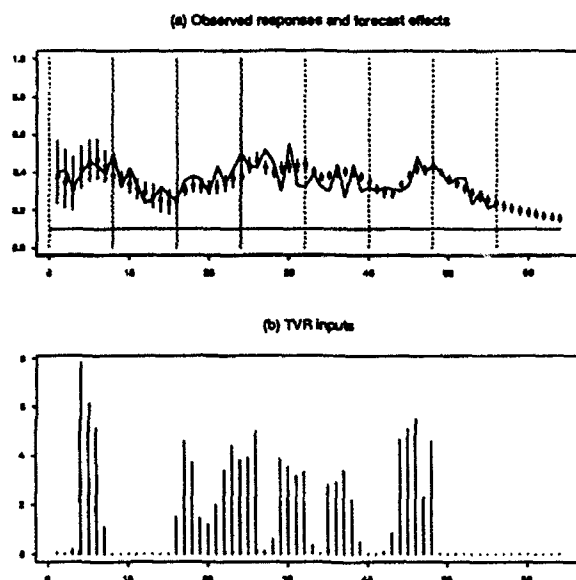


Figure 3. Advertising data and forecasts

However, to provide comparison, such a one-shot analysis was performed, using all the data at once. This analysis, along the lines of that in West (1992a), is simpler to perform, and is not subject to the possibility of approximation errors 'building-up' through the sequential evolution/ updating process. It is based on the same prior specification as above, and uses adaptive refinement of importance sampling functions through stages of sample sizes of 1000, 1500, 1500 and, finally, 4000. Again, at each stage, reduction is made to 500 components before proceeding. The resulting agreement with the sequential analysis is excellent. Final posterior means and standard deviations differ only in the *third* decimal place in each case, as does the estimated correlation between the transforms of ρ and κ . Correlation between the transforms of ρ and E_{56} is estimated, to two decimal places, at 0.94 rather than 0.95, that between the transforms of κ and E_{56} at -0.73 rather than -0.74 . These are differences of negligible practical import and easily attributable to the inherent randomness of the Monte Carlo.

References

- Andrade Netto, M.L., Gimeno, L., and Mendes, M.J. (1978) On the optimal and suboptimal nonlinear filtering problem for discrete time systems. *IEEE Trans. Aut. Con.*, AC-17, 439-448.
- Erkanli, A., Müller, P., and West, M. (1992) Curve fitting by a mixture of Dirichlet process model. *ISDS Discussion Paper #92-A09*, Duke University.
- Escobar, M.D., and West, M. (1992) Bayesian density estimation and inference using mixtures. Invited revision for *Journal of the American Statistical Association*.
- Geweke, J.F. (1989) Bayesian inference in econometric models using Monte Carlo integration. *Econometrica*, 57, 1317-1340.
- Kitagawa, G. (1987) Non-Gaussian state space modelling of nonstationary time series (with discussion). *J. Amer. Statist. Ass.*, 82, 1032-1063.
- Migon, H.S., and Harrison, P.J. (1985) An application of non-linear Bayesian forecasting to television advertising. In *Bayesian Statistics 2* (eds. J.M. Bernardo, M.H. DeGroot, D.V. Lindley and A.F.M. Smith) North-Holland, Amsterdam, and Valencia University Press.
- Müller, P. (1993) Posterior intergration in dynamic models (this proceedings).
- Pole A., and West, M. (1990) Efficient Bayesian learning in non-linear dynamic models. *J. Forecasting*, March issue.
- Silverman, B.W. (1986) *Density Estimation for Statistics and Data Analysis*. Chapman and Hall, Monographs on Statistics and Applied Probability. London and New York.
- West, M. (1981) Robust sequential approximate Bayesian estimation. *J. Roy. Statist. Soc. (Ser. B)*, 43, 157-166.
- West, M. (1990) Bayesian kernel density estimation. *ISDS Discussion Paper #90-A02*, Duke University.
- West, M. (1992a) Approximating posterior distributions by mixtures. *Journal of the Royal Statistical Society (Ser. B)*, 55, to appear.
- West, M. (1992) Modelling with mixtures (with discussion). In *Bayesian Statistics 4*, (eds: J.O. Berger, J.M. Bernardo, A.P. Dawid and A.F.M. Smith), Oxford University Press (to appear).
- West M., and Harrison, P.J. (1989) *Bayesian Forecasting and Dynamic Models*. Springer-Verlag, New York.

STATISTICAL REPRESENTATIONS AND ANALYSES OF SOFTWARE*

William M. Evanco
William W. Agresti

The MITRE Corporation
7525 Colshire Drive
McLean, Virginia

Abstract

This paper discusses the characteristics of multivariate statistical models for the analysis of software metrics. The choice of model is related to the data available, to the desired scale of the analysis, and to the purpose for which the model is used. Statistical models for the software quality factor of reliability are discussed. These models relate reliability to measures characterizing the software product and the characteristics of the development environment.

1.0 Introduction

Over the years, there have been a multitude of research activities aimed at measuring software development efforts. The utility of these activities is that software development organizations and government contract officers may learn how to better control the development process, and potentially improve the software product.

The quantitative characterizations of software development loosely fall into two classes: *product* metrics and *process* metrics. Product metrics may either be derived exclusively from the software product itself (e.g., design documentation or implemented code), or from various probes to the product (e.g., test programs generating defect counts).

On the other hand, process metrics focus on the development process and the decisions which are made during the conduct of the development project. These metrics include measures of software reuse, requirements volatility, and staffing effort devoted to various development activities.

A subset of the product/process metrics relating to software development outcomes has been the subject of study in the literature. These metrics relate to outcomes that development managers or government contract officers may want to track or predict. They include development effort, productivity, and various software quality factors such as reliability and maintainability. These metrics are treated as "explained variables," i.e., those whose values may be determined to some extent by one or more of the other process/product metrics. In this way, product/process details at some phase of the development effort could be used to predict software development outcomes at some future phase.

A modeling exercise determines the relationships of development outcomes to product/process metrics. The relationships are hypothesized, and tested with real world data. Carefully controlled experimentation of software development projects is not generally practical. Statistical analyses must be conducted on "naturally occurring experiments" without experimental controls. Multivariate statistical models are a tool for these kinds of analyses, and will be used in the analyses presented here.

This paper discusses the characteristics of these models and relates the choice of model to the data available, to the desired

* This research was supported by the Mission Oriented Investigation and Experimentation (MOIE) Program of The MITRE Corporation.

scale of the analysis (e.g. project, subsystem, or software module), and to the purpose for which the model is used (e.g. predictive or prescriptive).

2.0 Software Metrics Analysis as Empirical Science

An empirical science is one in which models are built to test theories or hypotheses related to measurable empirical reality. A model associates two or more variables with each other. The variables are operationally defined so that they can be empirically measured. The theory or hypothesis will be sustained if the model has explanatory or predictive power relative to the empirical world.

Empirical science is divided into experimental science and "historical" science. Experimental science allows for the conduct of controlled experiments for empirical data collection. If a relationship is posited between two variables, an experiment may be designed to fix the values of all other variables. In this way the systematic dependence of one variable on another may be established. Physics and chemistry are examples of experimental sciences.

On the other hand, there are fields of study qualifying as "historical" science. These include astrophysics and economics. With "historical" sciences it generally is not practical to conduct controlled experiments. The empirical data of astrophysics took place many eons ago and evidence of such events may just be arriving at earth (limited by the speed of light). Economics is often concerned with large socio-economic systems, and controlled experimentation may be either prohibitively costly or impractical.

Is software metrics analysis an "historical" or an experimental science? When software metrics research was first conducted in universities, it was conceived as an experimental science. Relatively controlled experiments were conducted on

student subjects writing computer code. For example, the effort needed to develop programs was related to measures of program complexity. But when dealing with human subjects, it is often difficult to completely control for other variables such as student intelligence or experience. However, a carefully designed statistical experiment can minimize the effects of these variables.

As interest in software metrics analysis moved to "real life" development projects, controlled experimentation became remote. The large size and complexity of these projects prohibited repeating them under controlled conditions. Thus, software metrics shifted toward being an "historical" science. Controlled experimentation was replaced by the collection of data from software artifacts and development process characteristics. The potentially obfuscating effects of other variables were controlled by analytical rather than by experimental means.

3.0 Software Development Outcomes

The assumption of most previous software metrics studies has been that product complexity influences software development outcomes such as reliability or maintainability [5]. Product complexity may be indicated, for example, by measures of lines of code or cyclomatic complexity [8]. Software with intricate processing relationships may be more difficult to implement correctly or test effectively, leading to lower reliability. Similarly, complex software may be more difficult for a maintainer to understand and modify correctly.

Early software metrics studies used a univariate approach to correlate development outcomes and complexity measures. Various complexity measures were compared (e.g., lines of code, cyclomatic complexity, Halstead measures [6]) for their abilities to explain development outcomes.

The emphasis in more recent software metrics studies [1, 4] has been to recognize the multi-dimensional nature of complexity. Two systems with identical numbers of lines of code can have greatly different cyclomatic complexities. If defects or maintenance effort were being predicted, then both lines of code and cyclomatic complexity might affect the outcome. Thus, the explanation of a software development outcome is a multivariate problem.

4.0 Levels of Analysis

Statistical analyses can be conducted at different levels of aggregation. The choice of level is often limited by the availability of data.

At the highest level, the data is expressed at the project level. Development outcomes are collected as project totals or project averages. The total number of defects identified during project testing is a project total. Effort per thousand lines of code (i.e., the inverse of productivity), and the average time to fix a defect are examples of project average metrics.

Project level data require consistent measurement definitions across projects. If different projects measure effort data in different ways, inter-project comparisons of productivity will be difficult. Statistical analyses require a multitude of observations exhibiting a variation in the explanatory variable(s). Collecting a large number of consistent cross-project data for statistical analyses can be difficult.

Consequently, lower data aggregation levels are attractive. Data may be collected for configuration items, subsystems, or "modules." Moreover, this data may be pooled across projects. For example, four projects with a total of thirty subsystems will yield thirty observations for subsystem level analysis rather than four observations at the project level. If desired, statistical models with lower levels of granularity, may be aggregated to higher levels.

Lower aggregation levels yield a broader base of observations, allowing for statistical models having more explanatory variables. The disadvantage of lower aggregation levels is the larger statistical fluctuations of the observations, which leads to lower levels of explained variation. At higher aggregation levels, these statistical fluctuations tend to smooth out.

For example, an important determinant of productivity at the module level may be the experience of the programmer. If this measure is not available, then this determinant will be ignored, and the explained variation may be lower. At the project level, the variations in average team experience across projects might be expected to be smaller than programmer experience across modules.

In the following sections, we focus only on subsystem level and "module" level analyses, and present results for software reliability analyses. The software project data and the associated development environment from which the data was extracted are discussed in [1, 2].

5.0 Subsystem Level Analyses

The subsystem level analyses are demonstrated for twenty-one observations from four Ada projects. The objective is to relate measures of software reliability to product and process measures collected in the design phase. Reliability, as the term is used in this paper, refers to the defects identified during the testing phase. The analyses discussed below use both total defects and defect densities as outcome variables.

The statistical models can be used as follows: Given product/process measures at the design phase, the software reliability as revealed in the testing phase can be predicted, and the relative contributions of each explanatory variable can be identified.

Such models have utility for a number of reasons. First, development managers

may want to know the expected number of defects in each of the subsystems in the testing phase. With this knowledge, test plan refinement and test resource allocations can be made.

Second, reliability models can provide prescriptive recommendations to improve the design or the development process. A large number of defects predicted for a subsystem may indicate that its design is too complex. In this case, action to simplify design can be taken in the design phase, using the reliability model to assess the impact of individual changes. If design characteristics are coupled (e.g., improving one worsens another), then the model can be used to analyze the net effect of design changes.

5.1 Software Defect Analyses

The results of statistical analyses treating defects as a dependent variable are shown in Figure 1. All the variables are entered into the regressions as logarithmic transforms. The numbers in parentheses are the standard errors associated with the coefficient estimates, and the R^2 is the adjusted coefficient of determination.

SLOC, measuring source lines of code, is an important determinant of subsystem defects. Larger subsystems tend to have more defects. Forty-six percent of the variation in defects is explained by lines of code alone. Based on the coefficient estimates for all four regressions shown in Figure 1, we cannot reject the hypothesis that defects trend linearly with source lines of code.

The remaining explanatory variables in Figure 1 are motivated by considerations of structural complexity in Ada designs as discussed at greater length in reference [1, 2].

Within the Ada programming language, context coupling allows one

library unit aggregation¹ to import visible declarations from another library unit aggregation. The extent of context coupling in a subsystem can be measured by the number of "with" clauses per library unit. We hypothesize that a larger value of this metric implies greater complexity since the library units (LUs) will be more highly coupled. All else being equal, a design with high context coupling will lead to more defects than one with low context coupling. The coefficient of this variable, "WITHS PER LU," shown in Model 2 of Figure 1 is positive and highly significant, supporting this hypothesis.

Explanatory Variables	Dependent Variable: DEFECTS			
	Model 1	Model 2	Model 3	Model 4
Constant	-3.6 [*] (1.7) ^{**}	-6.0 (1.3)	-6.3 (1.3)	-.65 (1.2)
SLOC	.81 (.19)	.97 (.14)	.96 (.13)	.95 (.13)
WITHS PER LU		.46 (.11)		
IMEXP			.47 (.09)	.50 (.09)
VISIBILITY				.27 (.18)
R^2 (adjusted)	.46	.71	.75	.77

* Coefficient estimate

** Standard error of estimate

Figure 1: Regression Results for Four Models With Defects as Outcome Variable

A more refined measure of context coupling complexity can be defined by recognizing that "with" statements are not all equal. The "withed" library units have different numbers of visible declarations. Those with more visible declarations should be given greater weights. Weighting the "with" clauses in proportion to the number

¹ A library unit aggregation is defined here as a library unit along with its associated secondary units.

of imported visible declarations gives the total imports within a subsystem. Setting the proportionality factor equal to the inverse of the total number of exports, the summation of the weighted "withs" is equal to the ratio of the imports to the exports denoted by IMPEXP.

Model 2 of Figure 1 shows the results when IMPEXP is used as a measure of context coupling. The associated coefficient is positive and highly significant, the explained variation is higher than for "WITHS PER LU".

The variable, "VISIBILITY," indicates the extent that the imported declarations are visible to the various compilation units composing a library unit aggregation. If a library unit is "withed" into a specification, the imported declarations are visible to the corresponding body and subunits (if any). When imported declarations are used only at the body level and below, then there will be unnecessary visibility at the specification level. The visibility variable is defined as the number of imports times the number of compilation units to which the imports are visible, divided by the number of imports. For a subsystem, the visibility variable is averaged over all library units in the subsystem.

The VISIBILITY variable enters with the expected positive sign and somewhat improves the coefficient of determination; but the coefficient has a large standard error and is not significant.

5.2 Software Defect Density Analyses

Since the coefficients in Figure 1 for the source lines of code measure are not significantly different from unity for any of the regressions, SLOC can be eliminated as an explanatory variable by recognizing that:

$$\log(\text{DEFECT DENSITY}) = \log(\text{DEFECTS}) - \log(\text{SLOC}/1000)$$

where the DEFECT DENSITY is expressed as defects per thousand lines of source code.

Figure 2 shows the statistical analyses treating the logarithm of the defect density as the dependent variable. The explanatory variables also enter as logarithmic transforms. In Model 1, the imports and the exports are entered as independent variables. They are both significantly different from zero, but not significantly different in value from each other. Recognizing that:

$$\log(\text{IMPEXP}) = \log(\text{IMPORTS}) - \log(\text{EXPORTS}),$$

the analysis for Model 2 uses only the import-export ratio with no decline in the value of the coefficient of determination.

Explanatory Variables	Dependent Variable: Defect Density			
	Model 1	Model 2	Model 3	Model 4
Constant	.65* (.96)**	.27 (.28)	-.04 (.35)	.65 (.36)
EXPORTS	-.49 (.10)			
IMPORTS	.45 (.12)			
IMPEXP		.48 (.09)	.51 (.09)	.27 (.11)
VISIBILITY			.26 (.18)	.05 (.16)
MODSPLU				.27 (.08)
R ² (adjusted)	.58	.58	.62	.76

* Coefficient estimate

** Standard error of estimate

Figure 2: Regression Results for Four Models With Defect Density as Outcome Variable

The visibility metric is introduced in Model 3. While it has the expected positive sign, the metric is not significant on the basis of its standard deviation. However, the visibility metric increases the explanatory

power as indicated by the coefficient of determination.

For Model 4, a process variable measuring the volatility of the software development environment is introduced. This variable is the average number of (non-defect) modifications per library unit, and is denoted by MODSPLU. A volatile development environment having large numbers of software changes contributes to the complexity of the development effort, and may lead to more software defects. MODSPLU enters with the expected sign, and also substantially contributes to the explanatory power of the equation.

The graphical comparison of the subsystem actual defect densities with the predicted for Model 4 is shown in Figure 3.

Additional product variables (e.g., subprograms per library unit) and process variables (e.g., reuse) were identified, but are not shown in the above analyses. With only twenty-one observations, the number of explanatory variables entering the regression analyses is limited. Moreover, the small number of observations led to substantial correlations among the various explanatory variables. More sophisticated analyses at the subsystem level will have to await the building of a larger subsystem database.

6.0 Library Unit Level Analyses for Defects

In order to circumvent the problems with subsystem level observations, analyses of defects were conducted at the library unit level. We restricted these analyses to library units consisting of a specification, a body, and possible subunits for a total of 290 observations. Such library units specify subprograms or tasks within them, and require a body and possible subunits for their implementation. Library units, for example, containing only types and objects require just a specification. Such library units are less complex, and may be expected to account for both fewer and less

consequential defects. For the data used in this analysis, seventy-eight percent of the defects occurred in library units having bodies and possibly subunits.

Although a defect count is an integer, at the subsystem level the relatively large number of defects may be approximated by a real number. But at the library unit level, with fewer defects per library unit, the discrete nature of the defects becomes apparent. Thus, appropriate statistical techniques for the analysis of defects at the library unit level must be employed.

In the following analyses, we use a discrete categorical dependent variable model as discussed by Ashford [3]. The library unit defect counts are classified into seven categories: 0, 1, 2, 3, 4, 5, and >5 defects. As with the subsystem level analyses, reliability measured by library unit defects depends upon complexity, and this complexity is a multidimensional variable.

We assume that a composite complexity measure can be defined as a linear combination of the different complexities, X_1, X_2, \dots, X_m . This composite complexity can be written as:

$$C^* = -a_0 - a_1 * X_1 - a_2 * X_2 - \dots - a_m * X_m + e \quad (1)$$

where the minus signs are chosen for convenience. The parameters, a_0, a_1, \dots, a_m , are to be estimated statistically, and e is a residual error term representing the difference between the estimated and actual composite complexity measures.

The composite complexity measure, C^* , is not observed directly. However, as this measure crosses thresholds the number of defects increases, and these defects are observed. Expressed mathematically,

$$\begin{aligned} \beta_{i-1} < C^* \leq \beta_i \\ \Rightarrow (i-1) \text{ defects if } i=1,2,\dots,6 \\ \Rightarrow > 5 \text{ defects if } i=7 \end{aligned} \quad (2)$$

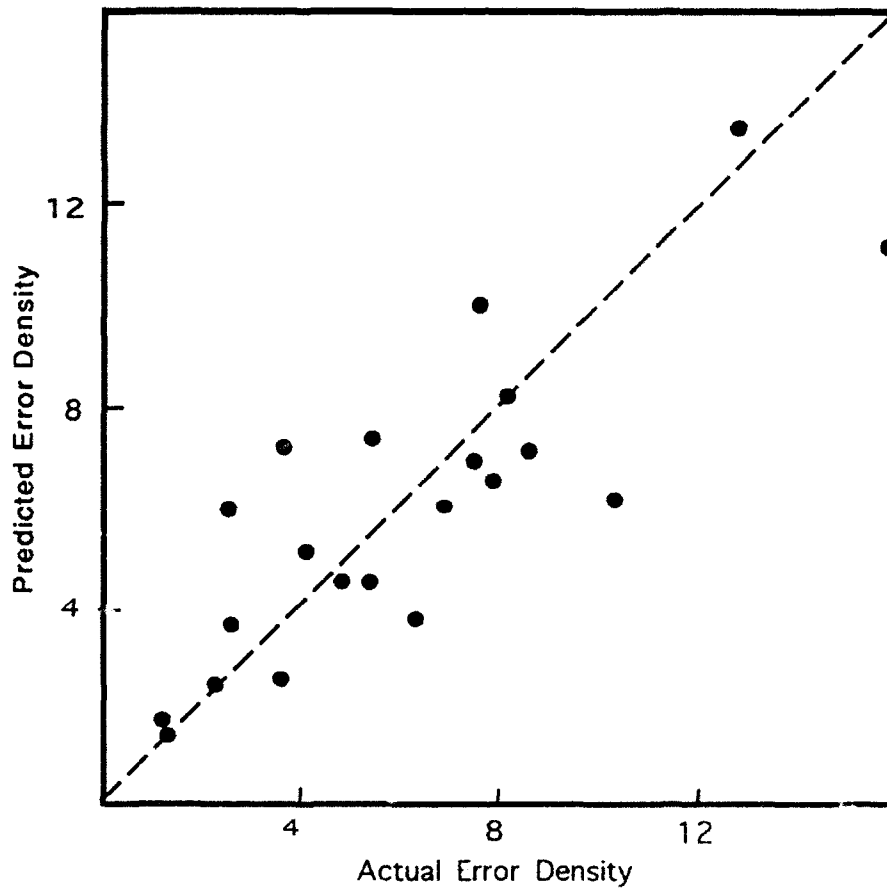


Figure 3: Predicted vs. Actual Error Densities
(Subsystem Analysis)

where the β_i represent the thresholds with $\beta_0 = -\infty$ and $\beta_7 = +\infty$ and $\beta_i < \beta_{i+1}$ for $i=1, \dots, 5$.

Substituting equation (1) into equation (2) and rearranging provides constraints on the residual, ϵ :

$$\begin{aligned} \beta_{i-1} + A \cdot X < \epsilon \leq \beta_i + A \cdot X \\ \Rightarrow (i-1) \text{ defects if } i=1, 2, \dots, 6 \\ \Rightarrow > 5 \text{ defects if } i=7 \end{aligned} \quad (3)$$

where the parameters and the variables are expressed as vectors:

$$\begin{aligned} A &= (a_0 \ a_1 \ a_2 \ \dots \ a_m) \\ X^T &= (1 \ X_1 \ X_2 \ \dots \ X_m) \end{aligned} \quad (4)$$

where X^T is the transpose of a column vector, X

Given a probability density function (PDF) for the residual, the probabilities of 0, 1, ..., 5, >5 defects in a library unit can be calculated. The PDF is arbitrary relative to scale and translation transformations, and may be chosen with unit standard deviation centered at the origin. It is denoted by the function $\text{PDF}(u)$ with corresponding cumulative distribution function, $\text{CDF}(u)$.

Thus far no assumptions have been made regarding the probability distribution of the residual, ϵ . If we assume that the residual is distributed normally, then the CDF is the error function, denoted by $\text{erf}(u)$, and the defect probabilities can be written as:

$$\begin{aligned} \text{Prob}(0 \text{ defects}) &= \text{erf}(\beta_1 + A \cdot X) \\ \text{Prob}(1 \text{ defect}) &= \text{erf}(\beta_2 + A \cdot X) \\ &\quad - \text{erf}(\beta_1 + A \cdot X) \\ \text{Prob}(2 \text{ defects}) &= \text{erf}(\beta_3 + A \cdot X) \\ &\quad - \text{erf}(\beta_2 + A \cdot X) \\ &\vdots \\ \text{Prob}(>5 \text{ defects}) &= 1 - \text{erf}(\beta_6 + A \cdot X) \end{aligned} \quad (5)$$

Note that the probabilities given in (5) sum to unity, the expected number of defects is given by:

$$\begin{aligned} \text{Exp(Defects)} &= \sum_{i=0}^5 i \cdot \text{Prob}(i \text{ defects}) \\ &\quad + \text{exp}(>5 \text{ defects}) \cdot \text{Prob}(>5 \text{ defects}) \end{aligned} \quad (6)$$

where $\text{exp}(>5 \text{ defects})$ is the average number of defects for those library units with more than five defects.

The results of the analysis are shown in Figure 4. The values of the β_i cannot be determined uniquely because of the translation invariance. Instead, what is reported is the sum of β_i and a_0 (the constant term of $A \cdot X$). These values are expected to increase monotonically with i based on the constraint equations (3).

Each explanatory variable is expressed so that a larger value implies greater complexity. Since we hypothesize that greater complexity leads to more defects, the associated parameters for all of the variables are expected to have negative signs. Since the probabilities sum to unity, the probabilities for larger defect numbers increase as the complexity variables increase while those for smaller defects decrease.

Aside from the variables used in the previous regression analyses, three new complexity measures were introduced. These are the:

- number of visible program units per library unit
- source lines of code per program unit
- parameters per program unit.

A larger number of visible program units may be expected to increase the implementation complexity, leading to more

Explanatory Variables	Model 1	Model 2	Model 3	Model 4	Model 5
$\alpha_0 + \beta_1$.41 (.09)	.41 (.11)	.83 (.14)	.86 (.16)	.95 (.17)
$\alpha_0 + \beta_2$.82 (.10)	.83 (.12)	1.26 (.14)	1.29 (.17)	1.43 (.18)
$\alpha_0 + \beta_3$	1.13 (.11)	1.13 (.12)	1.57 (.15)	1.60 (.17)	1.80 (.18)
$\alpha_0 + \beta_4$	1.36 (.12)	1.37 (.13)	1.83 (.16)	1.86 (.18)	2.10 (.19)
$\alpha_0 + \beta_5$	1.52 (.12)	1.53 (.13)	2.00 (.16)	2.03 (.19)	2.31 (.20)
$\alpha_0 + \beta_6$	1.69 (.13)	1.69 (.14)	2.19 (.17)	2.22 (.19)	2.54 (.21)
WITHS PER LU	-.07 * (.009)**	-.07 (.009)	-.04 (.009)	-.04 (.011)	-.03 (.009)
VISIBLE PROG UNITS		-.0008 (.006)	-.01 (.006)	-.01 (.005)	-.007 (.006)
SLOC PER PROG UNIT			-5.11 (1.03)	-5.09 (1.04)	-3.75 (1.07)
PARAMETERS/PROG UNIT				-.01 (.003)	-.05 (.03)
MODSPLU					-.09 (.01)
R^2	.23	.33	.44	.48	.51

* Coefficient Estimate

** Standard error of estimate

Figure 4: Results of Discrete Categorical Analyses--
Number of Defects in Library Unit

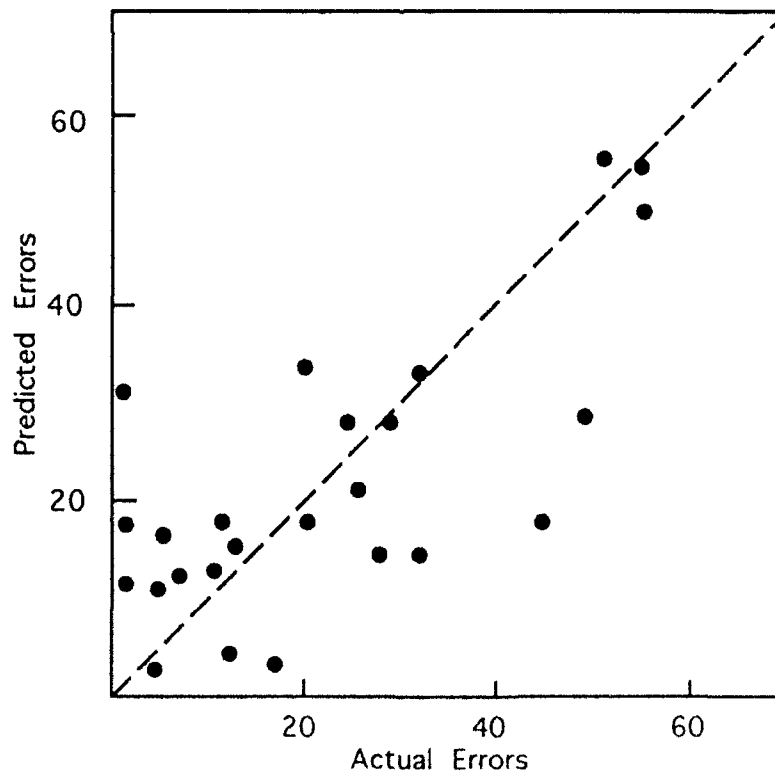


Figure 5: Predicted vs. Actual Errors (Library
Units Aggregated to Subsystems)

defects. Given two library units with the same number of visible program units, the implementation complexity may be greater for the library unit with more source lines of code per program unit leading to more defects. Finally, the number of parameters per program unit is a crude measure of the processing requirements. Greater processing requirements may lead to greater complexity and the potential for more defects.

The parameter values $\alpha_0 + \beta_i$, $i=1, \dots, 6$ increase monotonically with i as expected. The coefficients for all of the variables have the expected negative sign. They are significant to within the ten percent level of significance with the exception of the visible program units in Models 2 and 5, and the parameters per program unit in Model 5. The coefficients of determination are calculated from Maddala [7] using:

$$R^2 = \frac{1 - (L_\omega / L_\Omega)^{n/2}}{1 - (L_\omega)^{n/2}} \quad (7)$$

where n is the number of observations, L_ω is the likelihood function for estimators based only on the constants, while L_Ω is the likelihood function when the variables are also entered into the equation.

Equation (6) was used to calculate the expected defects by library unit. The library unit results were then aggregated to the subsystem level and the actual vs. predicted defects is shown in Figure 5. The correlation between actual and predicted defects is .85.

7.0 Conclusions

Based on the above analyses, the perspective that software complexity is a multidimensional problem is substantiated. No single complexity metric can completely characterize a software development project. Rather, the project can be represented by measures of the software product complexity and measures of the complexity associated with the development environment.

Many studies have been handicapped by a lack of data to conduct sophisticated statistical analyses. Relatively limited project level data was used, and cross-project comparability of various measures (especially those relating to outcomes like development effort or software defects) was questionable.

The analyses presented in this paper overcome these problems by focusing on subsystem and module level aggregations of data. However, even at the subsystem level, data limitations restrict the analyses. The twenty-one subsystem observations could not support a larger number of explanatory variables. Moreover, a number of proposed explanatory variables were found to be correlated (probably spuriously).

On the other hand, the library unit analyses, involving several hundred observations, did not exhibit the problems cited above. The analyses were limited, however, by the unavailability of certain software product and process data. For example, complexity associated with the functional call tree was not introduced because the software analyzer used could not extract this information. Similarly, process measures, such as the experience of the software development staff, were not taken into account because of the difficulty of collecting such data.

Although the data came from different projects, these projects were, nevertheless, conducted in a common, and relatively stable software development environment. This assured that the defect data was comparable across projects, and many process variables, such as software development methodology employed, were, in effect, held constant.

The future of software metrics, and their application to software development depend upon the collection of larger amounts of accurate and compatible data. From this perspective, it would be worthwhile to instrument software

development projects for the automated collection of much of this data, and to provide this data (assuring project anonymity) to qualified researchers in the software metrics field. Doing so requires some investment which would mean added costs to software development efforts. However, the payoffs are large in terms of the potential to influence and improve the process of software development.

REFERENCES

1. Agresti, W., W. Evanco, M. Smith, and D. Clarson (1990), "An Approach to Software Quality Prediction from Ada Designs," MITRE Technical Report, MTR-90W00135, MITRE Corporation: McLean, VA.
2. Agresti, W., W. Evanco, M. Smith (1990), "Early Experiences Building a Software Quality Prediction Model," Proceedings of the Fifteenth Annual Software Engineering Workshop, NASA/GSFC.
3. Ashford, J.R. (1959), "An Approach to the Analysis of Data for Semi-Quantal Responses in Biological Response," *Biometrics* 15, pp. 573-581.
4. Card, D. and W. Agresti (1988), "Measuring Software Design Complexity," *Journal of Systems and Software*, Vol. 8, No. 3, pp. 185-197.
5. Curtis, B. (1980), "Measurement and Experimentation in Software Engineering," *Proceedings of the IEEE*, 68,9, pp. 1144-1147.
6. Li, H. F., and W. K. Cheung (1987), "An Empirical Study of Software Metrics," *IEEE Transactions on Software Engineering*, Volume 3, Number 6, pp. 697-706.
7. Maddala, G. (1983), Limited Dependent and Qualitative Variables in Econometrics, Cambridge: Cambridge University Press.
8. McCabe, T. and C. Butler (1989), "Design Complexity Measurement and Testing," *Communications of the ACM*, Vol. 32, No. 12, pp. 1415-1425.

Parameter Estimation for Software Reliability Models Based on Delayed S-Shaped NHPP *

Amrit L. Goel and Kune-Zang Yang
Department of Electrical and Computer Engineering
Syracuse University, Syracuse, NY 13244.
and

Raymond Paul
US Army- OPTEC, Alexandria, VA, 22302

Abstract

Stochastic models based on the non-homogeneous Poisson process are increasingly being used for software reliability assessment. A major difficulty in their practical use is the estimation of model parameters which are obtained by numerical methods and are generally very sensitive to the initial values. In this paper we address this problem for the delayed S-shaped NHPP model but our results are also applicable to other popular NHPP models.

1 Introduction

An important measure of the quality of a software system is its estimated reliability at various points during system testing. A commonly used approach for determining the reliability is to fit an appropriate stochastic model to the available failure data and, based on this model, determine the current system reliability. Future reliability values are then obtained by judiciously extrapolating the fitted model. A commonly used model that has been found to be useful for this purpose is based on the non-homogeneous Poisson process (NHPP). It was originally proposed by Goel and Okumoto in 1979 [2]. Since that time, it has been employed in a variety of environments [3, 5, 7]. In addition to its simplicity, it has very good theoretical and practical interpretation. The original model was based on the exponential (NHPP-EXP) mean value function, and since then several modifications to the exponential form have been proposed by other authors. Two popular modifications are the delayed S-shaped (NHPP-DSS) and inflection S-shaped (NHPP-ISS) mean value

functions [3, 5]. Many investigators have found that at least one of these three mean value functions can be used to describe the failure process in most situations. In some cases, all three are applicable but one may give better results than others.

One major difficulty in the practical use of these reliability models is the estimation of the model parameters. In general, the estimation equations have to be solved numerically and the results tend to be very sensitive to the initial values chosen for the numerical procedure. In this study, we address this problem for the DSS model by first studying the characteristic points of the theoretical mean value function and its derivatives. Then we derive some relationships between the model parameters and the characteristic points. Finally, we use these relationships and the data derived trend test to develop guidelines for determining good initial values for the model parameters.

2 Delayed S-Shaped NHPP Model

The NHPP-DSS has a two parameter mean value function, $m(t) = a(1 - (1 + bt)e^{-bt})$, where a is related to the number of faults and b to the fault detection rate. Let y_i be the number of observed failures by time t_i , $i = 1, 2, \dots, n$. Then

$$P\{m(t_1) = y_1, m(t_2) = y_2, \dots, m(t_n) = y_n / a, b\} \\ = \prod_{i=1}^n \frac{\{m(t_i) - m(t_{i-1})\}^{y_i - y_{i-1}}}{(y_i - y_{i-1})!} \\ \times e^{-(m(t_i) - m(t_{i-1}))}. \quad (1)$$

Given observed values y_1, y_2, \dots, y_n , the log likelihood

*This work was partially funded by AIRMICS under contract No. DAKF 11-90-C-0022

of parameters a and b is easily obtained as

$$\begin{aligned} L(a, b) &\equiv L \\ &= \sum_{i=1}^n (y_i - y_{i-1}) \ln \{m(t_i) - m(t_{i-1})\} \\ &\quad - \sum_{i=1}^n \ln \{(y_i - y_{i-1})!\} - m(t_n). \end{aligned} \quad (2)$$

Taking the derivatives of L with respect to a and b and setting them equal to zero, we obtain

$$a = \frac{y_n}{1 - (1 + bt_n)e^{-bt_n}}, \quad (3)$$

and

$$\begin{aligned} at_n^2 e^{-bt_n} &= \sum_{i=1}^n (y_i - y_{i-1}) \\ &\quad \times \frac{t_i^2 e^{-bt_i} - t_{i-1}^2 e^{-bt_{i-1}}}{(1 + bt_{i-1})e^{-bt_{i-1}} - (1 + bt_i)e^{-bt_i}}. \end{aligned} \quad (4)$$

The estimate of b is given as one of the solutions of the following equation which is obtained from equations (3) and (4). Then a can be obtained by substituting for b in (3).

$$\begin{aligned} \frac{y_n t_n^2 e^{-bt_n}}{-(1 + bt_n)e^{-bt_n}} &= \sum_{i=1}^n (y_i - y_{i-1}) \\ &\quad \times \frac{t_i^2 e^{-bt_i} - t_{i-1}^2 e^{-bt_{i-1}}}{(1 + bt_{i-1})e^{-bt_{i-1}} - (1 + bt_i)e^{-bt_i}}. \end{aligned} \quad (5)$$

We have found that equation (5) may have multiple solutions and some of them could be misleading. We have developed an approach to overcome this estimation problem based on the trend test and some characteristic points of the observed data to find initial values for the model parameters.

3 Characteristic Points for an NHPP Model

There are three characteristic points which are useful in determining the initial estimates for model parameters. These are associated with the failure rate function $\lambda(t)$, where $\lambda(t) = \frac{d}{dt}m(t)$. These points are defined as follows.

K_1 : the time at which $\frac{d\lambda(t)}{dt}$ is maximal.

K_2 : the time at which $\lambda(t)$ is maximal.

K_3 : the time at which $\frac{d\lambda(t)}{dt}$ is minimal.

For the NHPP-DSS model, only K_2 is relevant while K_1 , K_2 and K_3 all play an important role in parameter estimation of other NHPP models. Now, for the DSS model,

$$\lambda(t) = \frac{d}{dt}m(t) = ab^2te^{-bt}$$

Taking the derivative of $\lambda(t)$ and equating it to zero, we get

$$ab^2(e^{-bt} - bte^{-bt}) = 0.$$

The solution for the above is $t = 1/b$ and from the definition of K_2 , we have $K_2 = 1/b$. On substituting $1/K_2$ for b in equation (5), the estimation problem reduces to finding the value of K_2 . The objective of this substitution is that, if an approximate value of K_2 can be determined directly from the observed data, it can be used as an initial value for the root finding procedure. This will ensure that the solution obtained is the desired one.

One difficulty with this, however, is that the observed data tend to be subject to much noise and it is not easy to determine where $\lambda(t)$ has attained a maximum, i.e. determining K_2 visually can be difficult. To overcome the difficulty, we use the Laplace trend test as described below.

The Laplace trend test of observed data is given by [4].

$$u(k) = \frac{\sum_{i=1}^k (i-1)z_i - \frac{k-1}{2}}{\sqrt{\frac{k^2-1}{12y_k}}}, \quad k = 2, \dots, n. \quad (6)$$

where $z_i = y_i - y_{i-1}$. We have found that a plot of $u(k)$ versus time k can be used to approximate the value of the characteristic point K_2 . In particular, if $u(k)$ has only one peak, then we use this value as K_2 . If $u(k)$ has multiple peaks, we use a weighted average as an estimate of K_2 for the root finding procedure. Although it is a heuristic approach, in many applications we have found that it yields very good results.

4 Illustrative Example

Consider the failure data given in Table 1. The data were obtained from the test reports of a medium size software system over a thirty month period. A plot of the Laplace trend factor is given in Figure 1. We note that this plot indicates three positive relative maxima

at (3, 2.74), (14, 22.91) and (25, 22.54). A weighted average of these yields an estimate of K_2 as

$$K_2 = \frac{2.74 \cdot 3 + 22.91 \cdot 14 + 22.54 \cdot 25}{2.74 + 22.91 + 22.54} = 18.52.$$

Using $b = 1/K_2 = 0.054$ as the initial value for numerically solving equation (5), we obtain $b = 0.046$. Substituting this in (3), we get $a = 4312.28$. A plot of the actual failure data and the fitted NHPP-DSS model is given in Figure 2.

It should be noted that, without using $b = 1/K_2$ as the initial value, a root $b \rightarrow 0$ was found and obviously was an unreasonable estimate of parameter b .

Table 1

Cumulative number of failures of a software system

Time	# of faults	Time	# of faults
1	20	16	740
2	40	17	780
3	80	18	850
4	90	19	920
5	95	20	920
6	100	21	1000
7	105	22	1150
8	110	23	1260
9	130	24	1460
10	150	25	1560
11	180	26	1640
12	280	27	1680
13	490	28	1700
14	650	29	1710
15	700	30	1720

References

- [1] D.R. Cox and P.A.W. Lewis, *The Statistical Analysis of Series of Events*. London: Chapman & Hall, 1978.
- [2] A.L. Goel and K. Okumoto, "Time-dependent error-detection rate model for software and other performance measures," *IEEE Trans. Rel.*, vol. R-28, no. 3, pp. 206-211, Aug. 1979.
- [3] S. Yamada and M. Ohba, "S-shaped reliability growth modeling for software error detection," *IEEE Trans. Rel.*, vol. R-32, no. 5, pp. 475-478, Dec. 1983.
- [4] H. Ascher and H. Feingold, *Repairable Systems Reliability: Modeling, Inference, Misconceptions and Their Causes (Lecture Notes in Statistics, Vol. 7)*, 1984.
- [5] M. Ohba, "Software reliability analysis models," *IBM J. Res. Develop.*, vol. 28, no. 4, July 1984.
- [6] A.A. Abdel-Ghaly, P.Y. Chan, and B. Littlewood, "Evaluation of competing software reliability predictions," *IEEE Trans. Software Eng.*, vol. SE-12, no. 9, Sept. 1986.
- [7] K. Kanoun, M.R. de Bastos Martini, and J.M. de Souza, "A method for software reliability analysis and prediction application to the TROPICO-R Switching System," *IEEE Trans. Software Eng.*, vol. 17, no. 4, April 1991.

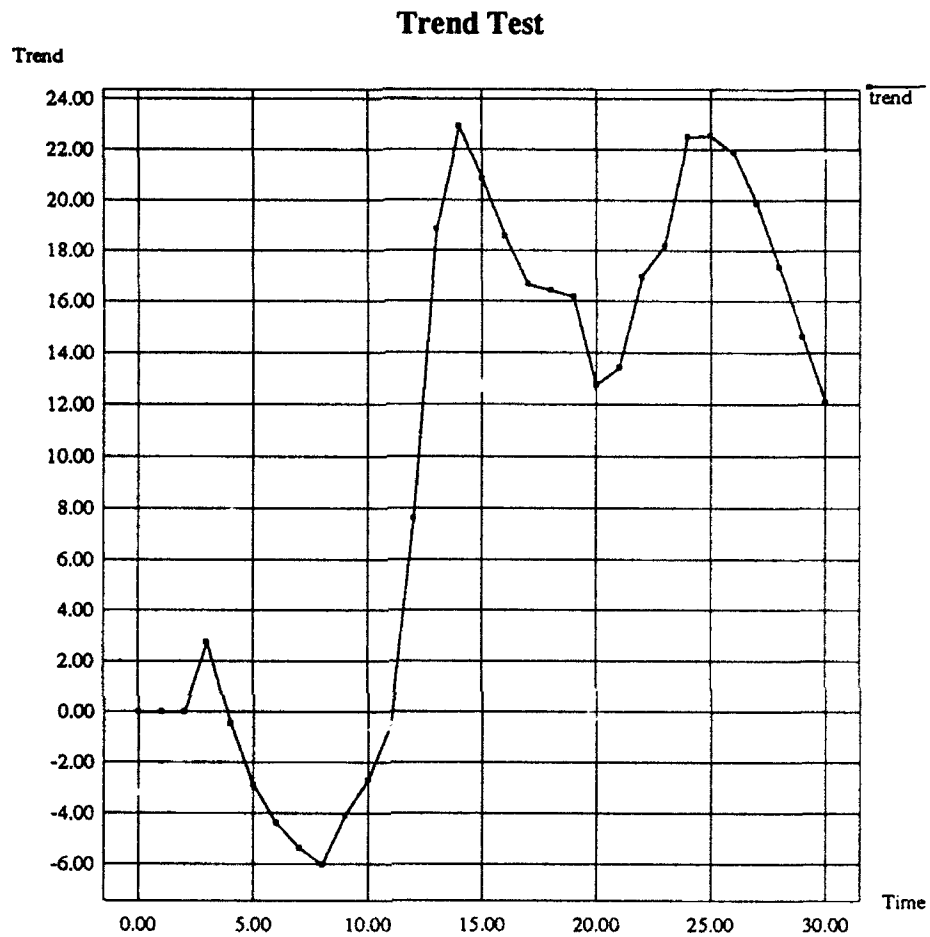


Figure 1: Laplace trend test

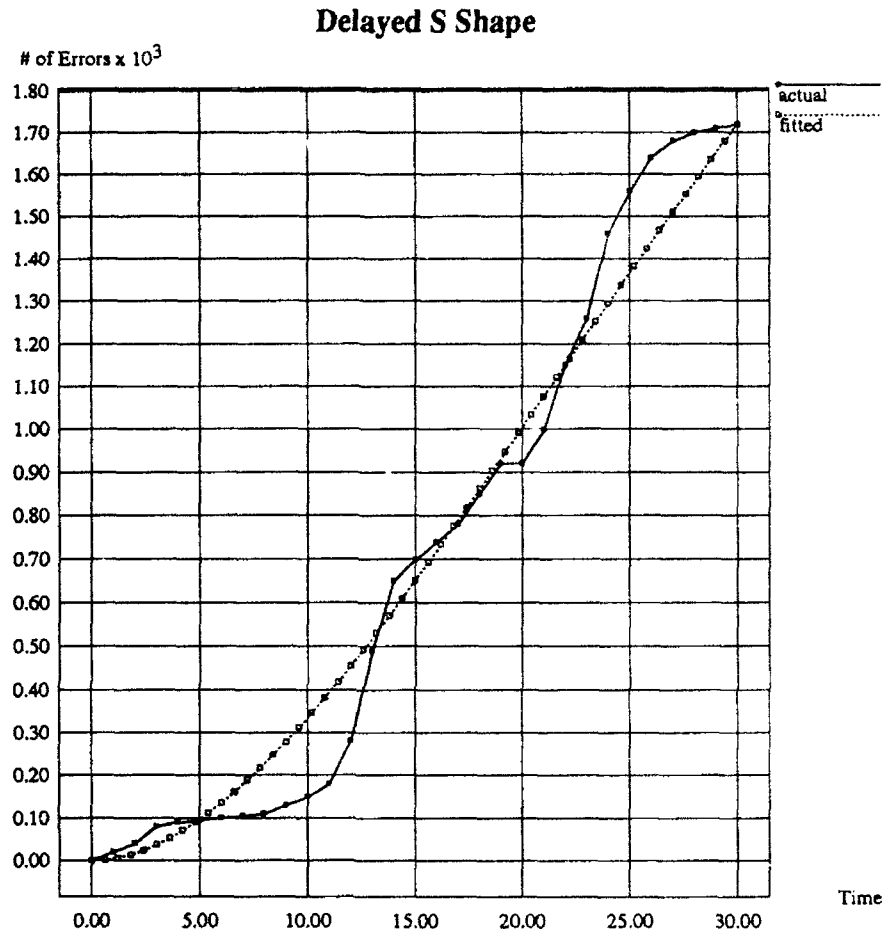


Figure 2: Failure analysis by Delayed S-Shape model

The Role of Statistical Reliability Assessment*

Andy Podgurski

Computer Engineering & Science Department
Case Western Reserve University
Cleveland, Ohio 44106

Abstract

The assessment and demonstration of software reliability is a controversial topic addressed by three principle approaches: software testing, program verification, and statistical inference. Although each of these approaches has apologists who dogmatically assert its primacy, none of the approaches has proven universally superior. Each approach is well-suited to some applications but not to others. The proper roles of software testing, program verification, and statistical reliability assessment vis-à-vis each other have not been established heretofore. We propose a delineation of these roles, based on an analysis of the relative strengths and weaknesses of each approach.

1 Introduction

As computers have pervaded our daily lives, we have become dependent upon computer software and vulnerable to harm caused by its failure. Such commonplace activities as using the telephone, driving, shopping, banking, and air travel now involve software. Although computers have often been portrayed as engines of mathematical certainty, precise and infallible, they and their software are often far from perfect. Software fails when it is invoked to carry out a computation which it was not correctly designed for. Many harmful software failures, leading to economic loss, injury, and even death, have been reported [Leve86, Neum92]. Software users are not the only victims of such failures. Those who sell software or products employing it are harmed by loss of sales and by legal liability for defective software [Salt89]. The dangers of software failures have led governments to investigate and even legislate the manner in which software is developed [UK88, US89].

Given the evident risks of software failure, one might

assume that responsible developers of software would not release their products without first assessing their reliability using methods of proven effectiveness. To the contrary, however, the methods most often used to assess software reliability are *ad hoc* and unsatisfactory even to those who apply them. Computer scientists have extensively investigated means to rectify this problem. Three principal approaches to software reliability assessment have emerged from these researches: **software testing**, **program verification**, and **statistical inference**. A program is tested by evaluating its behavior when it is executed on test data selected to reveal faults it might contain. A program is verified by logically deducing its correctness from a set of axioms. A program's reliability is assessed statistically based on the behavior the program exhibits when it is executed on a random sample of inputs from its operational environment. Unfortunately, there is no consensus about the utility of these three approaches. Debate between their proponents has often proven dogmatic and futile. Because of this controversy and other obstacles, the practice of software reliability assessment stagnates in an unacceptable state.

It is a startling fact that, despite these problems, the principal approaches to software reliability assessment are *satisfactory* in many cases. None of these approaches is universally superior to the others; however, each is well-suited to circumstances that the others are not. These facts are not widely recognized, due to the special nature of scientific inferences concerning program behavior. We endeavor to clarify this nature and to delineate the proper roles of software testing, program verification, and statistical reliability assessment *vis-à-vis* one another, by analyzing the strengths and weaknesses of each approach.

We argue in Section 2 that software testing is best suited to *subjective* reliability assessment, and should be used *intuitively* by someone with well-focused concerns about a program, in the manner of a mathematician seeking a counterexample to a conjecture. We contend

*Professor Podgurski's research was supported by NSF Research Initiation Award CCR-9009375

that many systematic testing methods are ill suited to subjective reliability assessment, because they are too abstruse to permit intuitive interpretation of testing results.

Because programs are artificial, some believe that inferences about them should be made *deductively*, as in program verification [Dijk76, Hoar69, Hoar86]. However, in Section 3 we present new arguments supporting the view expressed in [Fetz88, DeMi79] that purely deductive assessment of large software systems is an illusory goal. Nevertheless, program verification is appropriate for certain programs, which we describe.

Although statistical inference is objective and widely accepted in other fields, it is relatively neglected in software reliability assessment. In Section 4, we describe the reasons for this and argue that they are not well-founded. We describe the necessary conditions for statistical assessments of reliability to be predictive, and we describe a technique called *random input logging* for ensuring that these conditions are satisfied. Finally, we present a theoretical result from [Podg91] that clarifies the role of randomness in statistical reliability assessment by relating the accuracy of an assessment to the randomness of the input sample used, the size of the program being assessed, and the size of any correct implementation of the program's requirements.

2 Software Testing

By far the most commonly used method of assessing software reliability is testing. Software testing involves executing a program on selected inputs, evaluating the program's behavior, and appraising the program's reliability in light of the results. Often, test data is selected subjectively, based on the tester's intuitions. Systematic testing methods, on the other hand, incorporate objective rules for selecting test data. **Systematic** testing methods are distinguished on the basis of these rules. **Coverage-based** testing methods require that test data be selected that covers (executes) given program structures, such as statements, branches, control flow paths, or data flow relationships. For example, **statement coverage** is a coverage-based testing method that requires each statement in a program to be executed at least once during testing. **Fault-based** testing methods call for selecting test data that will reveal specific types of faults whose presence in a program is suspected. For example, **mutation testing** is a fault-based testing method in which mutants of a program to be tested are created by making small changes to the program [Budd80]. Test data is selected to distinguish the mutants from the original program. Thus,

if the original program contains a fault that a mutation corrects, the fault will be revealed during testing. **Specification-based** testing methods call for exercising certain aspects of a program's requirements specification. For example, **functionality testing** involves attempting to exercise each type of functionality that a program is required to provide.

Reliability assessments based on testing are generally *subjective*, even when test data is selected systematically. There are significant obstacles to making objective reliability assessments via testing. For starters, it is known that no algorithm can decide whether an arbitrary program is correct or not [Loec84]. Furthermore, although testing may be viewed as a form of sampling, statistical sampling theory cannot be used to make inferences based on the results of either subjective or systematic testing, because neither approach calls for selecting test data randomly. It is plausible that one might demonstrate experimentally that a particular relationship typically holds between the reliability of a program and the results of testing it with a particular method. For example, one might demonstrate that with a certain testing method, the frequency of failures during testing is usually an upper bound on the long term failure frequency. However, no strong results of this type are known.

Objective reliability assessments are obviously better-suited for communication than subjective ones, particularly for contractual purposes. Nevertheless, well-founded subjective assessments are valuable for some purposes. Testing can sometimes be used to make such assessments more cheaply than objective assessments can be made, as we now describe.

A programmer's confidence in the reliability of their program is based on knowledge of how the program works and of how it will be used. Testing can be alleviate concerns about how a program works in two ways. The first is an incipient form of statistical inference. The tester selects test data they perceive to be "representative" operational inputs. If no failures are observed during testing, the tester infers that few failures will occur during operational use of the program. We defer further consideration of statistical reliability assessment to Section 4. The second way that concerns about a program can be alleviated by testing is very similar to the way that a mathematician comes to believe an unproven conjecture. Fetzner has likened a program and an execution of it to a *conjecture* and to a *refutation*, respectively [Fetz88]. We believe this comparison holds the key to understanding the proper role of software testing.

In mathematics, conjectures arise from various sources, including knowledge of proven theorems, gener-

alization from examples, and, especially, intuition. The author of a conjecture is not necessarily confident of its truth. If a conjecture is difficult to prove, strong doubts may arise. This may lead a mathematician to try to refute the conjecture, perhaps by providing a counterexample. If this succeeds, the conjecture must be modified or abandoned. Often, however, an attempt to refute a conjecture provides new evidence for its truth¹. A failed refutation may even suggest a way of proving the conjecture. It is not uncommon for a mathematician to alternate repeatedly between attempting to prove a conjecture and attempting to refute it, learning a little more from each attempt. A mathematician's attempts to refute a conjecture are not haphazard. They are guided by knowledge and intuition to investigate particular circumstances satisfying the hypothesis of the conjecture. This suggests a paradigm for software testing.

If a tester is profoundly confused about how a program actually works, they cannot make useful inferences about its reliability without resorting to something resembling statistical inference. If, on the other hand, they largely understand the program but have certain well-focused concerns about its behavior, they are in a position to judiciously select test data in order to explore the basis for those concerns. If a tester believes that a program can fail only in certain specific circumstances, they can select test data that gives rise to those circumstances and observe the results. If failures are observed, their doubts are confirmed; however, if no failures occur, their doubts may be exorcised. In this paradigm, tests are essentially proposed *counterexamples* to the conjectured reliability of a program. Thus, we call this the *conjecture-counterexample paradigm* for software testing².

The author of a program is best prepared to conduct this type of testing. They have extensive knowledge about the program's development that can be used to guide testing. They know what parts of the program they found difficult to write and which parts they found easy. They know what kinds of mistakes they typically make. They have some understanding of the consequences of different types of failures. When a programmer decides that they wish to test a certain aspect of their program, their knowledge of how the program works is invaluable in selecting appropriate test data. For example, a programmer may have no difficulty se-

lecting test data to execute a certain program statement, although no algorithmic method can do this in general.

Let us consider a simple example of how testing might proceed according to the conjecture-counterexample paradigm. The developer of a compiler for an object-oriented programming language is confident that their compiler is largely correct. This confidence is based on the compiler-writer's past experience and on their knowledge of the compiler in question. However, they are concerned about the garbage collection algorithm used by the new compiler to reclaim dynamic storage in object programs. The compiler writer had never implemented garbage collection before and found the problem subtle and difficult. In particular, they are concerned that the storage for objects that become unreferenced may not be collected under certain circumstances. They test the compiler using source programs that give rise to the circumstances in question, and observe, using a debugger, that the object programs reclaim storage correctly when executed. This helps convince the compiler-writer of the reliability of their compiler.

It is important to note that some systematic testing methods are incompatible with the conjecture-counterexample paradigm for software testing. Whereas the paradigm calls for selecting test data in an intuitively meaningful way, to explore well-focused concerns about a program, the systematic testing methods in question call for exhaustively exercising certain generic aspects of programs. An example of such a systematic testing method is path coverage, which calls for exercising every control flow path in a program. It is difficult to imagine how a tester could intuitively comprehend the results of path coverage, since the method is essentially divorced from the tester's knowledge and intuition about the program to be tested. The view that many systematic testing methods do not lend themselves to intuitive assessment of reliability explains the often-heard question (lament?) of practitioners, "If I test with method X and no failures occur, what do I know about my program?"

3 Program Verification

In principle, *deductive inference* suffices to completely explain a computer program's behavior. This is because the basic mechanisms of programs are accessible and well-understood. Hence, it is natural to consider using deductive methods to assess software reliability. In program verification, one seeks a deductive proof that a program is correct, that is, that the program will *never* fail. Obviously, such a proof is an ideal characteriza-

¹For example, the main support for the $P \neq NP$ conjecture of theoretical computer science is the failure of many attempts to devise polynomial-time algorithms for NP -complete problems [Gare79].

²Note that the conjecture-counterexample paradigm for software testing does not require that testing be used hand-in-hand with program verification, although that is natural to consider.

tion of reliability, once obtained. Proponents of program verification prefer the seeming certainty of deduction to the inherent uncertainty associated with *inductive inference*. The latter is used extensively in the natural sciences, because the basic mechanisms of many natural systems are poorly understood and because the complexity of natural systems often precludes purely deductive explication. Ironically, many programs exist now whose complexity rivals that of natural systems and renders the use of program verification problematic.

Program verifications may be either *formal* or *informal*. An informal verification yields a proof like those typically written by mathematicians. The proof is intended to be understood readily by humans and so is expressed in a mixture of natural language and mathematical notation. Rules of inference are applied implicitly. The proof may omit deductions that its author judges the reader can easily supply. A formal verification, on the other hand, yields a complete deduction within a formal proof-system. This consists of a sequence of statements expressed in a formal language. Each statement is either an axiom of the system or follows from earlier statements in the sequence by an explicit application of a rule of inference. The last statement in the sequence asserts that the program in question is correct. Formal proofs tend to be *much longer than informal ones*, because the latter are much less detailed than the former. A formal proof of a program's correctness contains an explicit deduction for every operation in the program.

Attempted proofs of correctness, whether formal or informal, are subject to flaws much like those that occur in programs. This problem is acute with large programs, for which correctness proofs are extremely long and complex. Manually constructing and checking such proofs is a laborious and error-prone task. Although informal proofs are shorter than formal ones, this advantage is weakened by an element of subjectivity inherent to their evaluation. Formal proofs, while objective, are so detailed as to often defy human comprehension. DeMillo, Lipton, and Perlis argue that proofs of program correctness are of dubious value because they are not likely to be subject to the kind of communal scrutiny that important proofs in mathematics are [DeMi79]. Fetzer asserts that while *algorithms* can be proven correct with respect to abstract models of computation, *programs* written to execute on physical machines cannot be proven correct, because unknown factors can affect program behavior. In any case, many software failures are caused by inappropriate requirements specifications, and program verification does not address this problem at all.

One potential advantage of formal proofs of program

correctness is that such proofs can be checked *automatically*. Formal proofs that are annotated to indicate what axioms, rules of inference, and previously proved statements are used at each step can be checked rather efficiently by a simple program. This doesn't, of course, obviate the need to construct a formal proof of correctness in the first place. Although some advocates of program verification have suggested that with training a programmer can manually derive formal proofs of correctness by "calculating" [Dijk89], there are fundamental impediments to such calculative verification³. Firstly, long calculations are error-prone — indeed, that is why computers were invented! Secondly, humans calculate using manual *algorithms*, but the question of whether an arbitrary program satisfies a specification is not even *decidable* algorithmically. Automated verification is an example of the *word problem* for rewriting systems, which has proven intractable in all but its most restricted forms [Benn87].

It is plausible that *interactive theorem proving*, in which a human interacts with an automated theorem-prover, might become a very useful approach to program verification [Rush91]. Will interactive theorem-proving eventually permit the "programmer in the street" to verify large programs in a timely fashion? This seems unlikely. Interactive theorem-proving requires considerable expertise with formal logic and automated deduction. The user of an interactive theorem-prover must guide it by suggesting lemmas leading to the goal-theorem. Thus, the user must be a skilled theorem-prover himself and must thoroughly understand the capabilities and limitations of the automated prover. Worse, to verify typical programs the user would need to be a proficient logician. The formal analogue of a programmer-defined data type is an *abstract data type* [Wirs90]. The definition of a particular abstract data type contains statements of formal logic that characterize the semantics of the type. Ensuring that these statements appropriately characterize required behavior is a problem of the greatest subtlety. In the history of mathematics there are many examples of controversies among great mathematicians about the proper constitution of axiom systems⁴ [Klin90]. Formal verification of software with programmer-defined data types requires a human verifier to supply data-type axiomatizations in bulk!

³An auxiliary benefit of "intuitionistic" program verification is that even a flawed attempt at verification may help to ensure that the form of a program is *nearly* appropriate, by enhancing the verifier's understanding of the program. This benefit is not realized with purely calculative, formal verification.

⁴Perhaps the most famous of these is the controversy surrounding Euclid's "Parallel Postulate".

Despite the aforementioned limitations of program verification, it is regularly and successfully used in some domains. In the area of theoretical computer science known as the *Design and Analysis of Algorithms*, informal verification is the primary vehicle for demonstrating the correctness of algorithms. Research papers that present an algorithm usually include an informal proof of its correctness. Of course, theoretical computer science is much like the mathematics in that the ultimate acceptance of a proof depends on a "social process" à la DeMillo *et al.* Moreover, the algorithms in question are usually rather small, albeit intricate. The cost of verifying them is justified by their fundamental nature, and unlike most software, these algorithms are not subject to frequent modification. Any software sharing these properties is a good candidate for verification.

When possible, it is appropriate to verify the software in *critical systems*, because failures of such software may lead to catastrophic consequences, such as loss of life [Leve86]. However, the software for some critical systems is too complex to permit effective verification. Leveson has argued that the benefit of some of these systems may outweigh their risk [Leve92b]. We contend that, otherwise, the construction of such systems *should not even be attempted* unless a *simple mechanism* can be identified that is independent of the primary software and that ensures the safe, if not completely correct, behavior of the overall system. Such a mechanism need not involve software at all, although it is a good candidate for verification if it does. An example of a simple fail-safe mechanism is the potentiometer eventually used to prevent the Therac-25 radiation-therapy machine from delivering dangerous doses of radiation [Leve92a]. Other examples of such mechanisms are discussed in [Dunn90].

4 Statistical Reliability Assessment

The statistical approach to software reliability assessment has some significant advantages over software testing and program verification. Unlike testing, it yields *objective* reliability assessments. The cost of statistical reliability assessment does not necessarily increase substantially with program size, as does the cost of verification. However, the cost of statistical reliability assessment is not insignificant. Moreover, statistical assessments are not certain, although the degree of uncertainty associated with them can be characterized probabilistically. When it comes to assessing software reliability, there is, apparently, no free lunch.

There are several ways to characterize reliability statistically, and there are several statistical procedures that can be used to assess reliability, e.g., [Brow75, Cho87, Goel85, Thay78]. In order to illustrate the basic issues of statistical reliability assessment, we present one simple but useful formulation of the problem. Suppose that we wish to assess the reliability that a deterministic program P exhibits when it is executed repeatedly in an environment E during a time period $[t_1, t_2]$. We might characterize P 's reliability as the proportion θ of P 's runs that succeed. Let $R = \{r_1, r_2, \dots, r_N\}$ be the set of runs of P that occur in E during $[t_1, t_2]$. Associate a binary variable y with these runs as follows: if run r_k succeeds then the value y_k of y associated with r_k is one; if r_k fails then $y_k = 0$. The proportion θ is of course the population mean $\bar{y}_R = 1/N \sum_{k=1}^N y_k$.

Usually, one wishes to *predict* θ for some future time period $[t_1, t_2]$. Unfortunately, it is rarely possible to determine how a program will be invoked in the future. Thus, θ cannot be determined or estimated directly. Instead, P is executed in an environment E' similar to E and P 's behavior is observed in order to approximate θ . Suppose that P is executed N' times in E' during a time period $[t'_1, t'_2]$. Let $R' = \{r'_1, r'_2, \dots, r'_{N'}\}$ be the corresponding set of runs of P , and let y'_k be one if run r'_k is successful and zero otherwise. It is assumed that the long-term relative frequency of the different inputs to P is approximately the same in E' as in E . Thus, if the intervals $[t_1, t_2]$ and $[t'_1, t'_2]$ are sufficiently long, then it is likely that $\theta \approx \bar{y}_{R'} = 1/N' \sum_{k=1}^{N'} y'_k$, because the behavior of P is determined by its inputs. The number N' of runs in R' is normally too large to permit the population mean $\bar{y}_{R'}$ to be calculated directly, because the success or failure of a run must usually be determined manually. However, $\bar{y}_{R'}$ can be assessed statistically by randomly sampling from R' . For example, we can test the hypothesis H_0 that $\bar{y}_{R'} \leq 0.999$, based on the number X of successful runs that are observed in a simple random sample of 3000 runs drawn with replacement from R' . The random variable X has a binomial distribution with parameters 3000 and $\bar{y}_{R'}$. It is not difficult to show that the procedure that rejects H_0 (accepts the hypothesis that $\bar{y}_{R'} > 0.999$) if $X = 3000$ is a uniformly most powerful test [Case90] at the level of significance $0.999^{3000} = 0.0497 \dots$.

The flexibility of software makes it possible for a program to facilitate its own reliability assessment. This is achieved by instrumenting the program to **randomly log** its inputs. At the beginning of each run, the program pseudorandomly generates a real number r from an (approximate) uniform distribution on the interval $[0, 1]$. The program then tests whether r is less than a

fixed logging probability π . If so, the program records all elements of its input in a permanent file, called the log; if $r \geq \pi$, the program does not record its input. In either case, the program carries out its normal computation. After the program has run many times, its log contains a random sample of operational inputs, which can be used to reinvoke the system during reliability assessment in order to evaluate the outcomes of a random sample of runs. In practice, random input-logging would be employed in a program's intended operating environment or in an approximation to it. If a program is intended to operate in many environments, then the technique can be applied over a random sample of them to study variations in reliability across environments.

The predictiveness of statistical reliability assessment depends on the degree to which the usage of a program during reliability assessment resembles its operational usage. Random input-logging in the operational environment permits current usage to be characterized faithfully. However, some software engineering researchers contend that operational usage can be quite changeable [Ham87]. There is no doubt that individual uses of a program can vary considerably. Nevertheless, myriad applications of statistics have demonstrated that variation between individuals does not preclude regularity in an aggregate. If a program's operational environment seems unlikely to change dramatically, it is reasonable to hypothesize that the long-term failure frequency of the (unmodified) program will either be stable or change only gradually. Such a hypothesis requires confirmation and periodic reconfirmation, although this is unusual in current practice. Random input-logging provides a convenient mechanism for collecting the data necessary to conduct period reassessments of software reliability.

Many programmers are extremely resistant to the idea of statistical assessment of software reliability. To them, random selection of test data seems hopelessly naive compared to thoughtful, purposive selection. This attitude frequently indicates a confounding of the role of software testing in reliability assessment with its role in *fault removal*. Testing is often used to find faults so that they can be corrected to *improve* reliability. In fact, testing is used for reliability assessment and fault removal simultaneously. Whereas objective reliability assessment calls for the selection of "representative" sample inputs, fault removal calls for selecting inputs that are likely to reveal latent faults. It is therefore plausible that subjective or systematic selection of test data is appropriate for fault removal, although such test data selection is not appropriate for statistical reliability assessment. Few programmers recognize this subtle but critical distinction.

Some authors contend that statistical methods are incapable of demonstrating **ultra-high reliability**, by which is meant a long-term failure frequency of one or fewer failures per billion runs [But91]. They argue that such a demonstration would require evaluating a program's behavior over billions of inputs, which is impractical⁵. This argument is compelling with respect to programs whose behavior must be evaluated manually. However, if a program's behavior can be evaluated *automatically*, then demonstration of ultra-high reliability is possible, provided that the relative frequency of different inputs does not change. For example, the output of a program to compute the square-root of a number can easily be checked by squaring it and comparing the result to the program's input. Thus, such a program's reliability can be evaluated automatically over a massive sample of inputs, permitting the demonstration of ultra-high reliability.

The reservations of many programmers about statistical reliability assessment bespeak an unfamiliarity with the crucial role of randomness in statistical inference. We close this section by stating a theoretical result that helps to clarify that role in the context of software reliability assessment [Podg91]. This theorem relates the accuracy of a statistical reliability assessment to the randomness of the input sample used and to the nature of the program assessed. In the theorem, the randomness of a sample is characterized in terms of **Kolmogorov complexity** [Li90]. Strictly speaking, Kolmogorov complexity is a measure of the randomness of strings (finite sequences of characters). However, it can be used to measure the randomness of a sample from a finite population by equating this with the Kolmogorov complexity of the **characteristic string** associated with the sample. The latter string is the binary string whose length is equal to the size of the population from which the sample was drawn and whose i th bit is one if and only if the i th element of the population belongs to the sample.

Intuitively, a string is random if it has *no simple pattern* or if, put another way, there is *no succinct description* of the string. The definition of Kolmogorov complexity is a formal characterization of this intuition. Informally, the Kolmogorov complexity $K(x)$ of a string x is the length of the shortest program that generates x . Note that a string x can always be generated by a program that actually contains a copy of x , so $K(x) \leq |x| + c$ for some constant c that is independent of x .⁶

⁵Others suggest that ultra-high reliability requirements are unrealistic [Litt91].

⁶Note that we denote the length of a string x by $|x|$ and the cardinality of a set S by $|S|$, relying on context to make clear the

Informally, our result states that if a statistical reliability assessment of a program's reliability is grossly misleading, then either: the input sample used was not very random, the program is very long, or every correct implementation of the program is very long.

Theorem 1 ([Podg91]) *Let S be a nonempty subset of $[N] = \{1, 2, \dots, N\}$ for some $N \geq 16$, and let x be the characteristic string of S with respect to $[N]$. Suppose that P and P' are programs that compute total functions g_P and $g_{P'}$, respectively, from $[N]$ to R for some set R . Then*

$$K(x) \leq |P| + |P'| + 2(\delta + \epsilon + 1)\lceil \log_2(N + 1) \rceil + c,$$

where $\delta = |\{i \in S : g_P(i) \neq g_{P'}(i)\}|$, $\epsilon = |\{i \in [N] - S : g_P(i) = g_{P'}(i)\}|$, and c is a constant that is independent of P , P' , N , and S .

Suppose that P is a program whose reliability we wish to assess, and suppose that P' is any program satisfying the requirements for P . Then δ is the number of times that P fails on the sample S , and ϵ is the number of inputs in $[N] - S$ for which the output of P is correct. Theorem 1 implies that if we assess the reliability of P based on the value of δ , but this assessment is grossly misleading (overly optimistic) in the sense that δ is a small fraction of N but $N - |S| - \epsilon$ is a large one, then, for large enough N , either S is not very random or the sum of the lengths of P and P' cannot be much smaller than N . This is because in these circumstances the term $2(\delta + \epsilon + 1)\lceil \log_2(N + 1) \rceil$ is negligible relative to N . (Note that the method used to prove Theorem 1 can easily be adapted to prove an analogous result about overly *pessimistic* reliability assessments.) It is not surprising that a reliability assessment obtained by executing a program on a nonrandom sample of inputs may be misleading; that is conventional statistical wisdom. However, Theorem 1 implies that if an assessment of a program's reliability is sufficiently misleading, if the program is not too long, and if there is a correct implementation of its requirements that is not too long, then the sample *cannot* be random. Theorem 1 can be extended to apply to programs P with non-integer inputs and to samples drawn from arbitrary operational distributions [Podg91].

5 Conclusion

We have seen that software testing, program verification, and statistical methods each have a useful role

intended meaning of the notation.

in software reliability assessment. At present, software testing permits only subjective reliability assessment. However, it may be the method of choice when a tester largely understands a program but has a few well-focused concerns about aspects of its behavior, because testing can be used intuitively to explore those concerns in the manner of a mathematician seeking a counterexample to a conjecture. However, some systematic testing methods are incompatible with this *conjecture-counterexample paradigm* for testing, because they select test data in a manner that is divorced from a tester's intuitions about a program. Program verification is an ideal method of reliability assessment when a program's requirements are well-understood and when the program itself is relatively small; however, it is ill-suited to large programs with highly contingent requirements. Interactive verification with the aid of an automated theorem-prover may extend the applicability of formal verification, but it is likely to remain a specialized tool. Statistical inference is a neglected tool for software reliability assessment, which can provide objective reliability assessments at relatively low cost. The predictiveness of statistical reliability assessment requires that the usage of a program during its assessment be very similar to operational usage. *Random input-logging* can be used to ensure that this condition is met. However, statistical demonstration of *ultra-high reliability* does not seem practical, except when the correctness of a program's behavior can be checked automatically.

References

- [Benn87] Benninghofen, B., Kemmerich, S., and Richter, M. M. *Systems of Reductions*. Springer-Verlag, Berlin, 1987.
- [Brow75] Brown, J. R. and Lipow, M. Testing for software reliability. *Proceedings of the International Conference on Reliable Software* (Los Angeles 1975), pp. 518-527.
- [Budd80] Budd, T. A., DeMillo, R. A., Lipton, R. J., and Sayward, F. G. Theoretical and empirical studies on using program mutation to test the functional correctness of programs. *Proceedings of the 7th Annual Symposium on Principles of Programming Languages* (Las Vegas, 1980), ACM Press, New York, 1980.
- [Butl91] Butler, R. W. and Finelli, G. B. The infeasibility of experimental quantification of life-critical software reliability. *Proceedings of the ACM SIGSOFT '91 Conference on Software for Critical Systems* (New Orleans, 1991), ACM Press, New York, 1991, pp. 66-76.
- [Case90] Casella, G. and Berger, R. L. *Statistical Inference*. Wadsworth and Brooks/Cole, Pacific Grove, CA, 1990.
- [Cho87] Cho, C. *Quality Programming*. Wiley, New York, 1987.
- [DeMi79] DeMillo, R. A., Lipton, R. J., and Perlis, A. J. Social processes and proofs of theorems and programs. *Communications of the ACM*, Vol. 22, No. 5 (May 1979), pp. 271-280.
- [Dijk76] Dijkstra, E. W. *A Discipline of Programming*. Prentice-Hall, Englewood Cliffs, NJ, 1976.
- [Dijk89] Dijkstra, E. W. On the cruelty of teaching computer science. *Communications of the ACM*, Vol. 32, No. 12 (December 1989), pp. 1398-1404.
- [Dunn90] Dunn, W. R. and Corliss, L. D. Software safety: a user's practical perspective. *How Safe is Control Software*, NASA Tech Brief ARC-12720.
- [Gare79] Garey, M. R. and Johnson, D. S. *Computers and Intractability: A Guide to the Theory of NP-Completeness*, Freeman, 1979.
- [Goel85] Goel, A. L. Software reliability models: assumptions, limitations, applicability. *IEEE Transactions on Software Engineering*, Vol. SE11, No. 12 (December 1985), pp. 1411-1423.
- [Haml87] Hamlet, D. Probable correctness theory. *Information Processing Letters*, Vol. 25, pp. 17-25 (April 1987).
- [Fetz88] Fetzner, J. H. Program verification: the very idea. *Communications of the ACM*, Vol. 31, No. 9 (September 1988).
- [Hoar69] Hoare, C. A. R. An axiomatic basis for computer programming. *Communications of the ACM*, Vol. 12, No. 10 (October 1969), pp. 576-583.
- [Hoar86] Hoare, C. A. R. Mathematics of programming. *BYTE* (August 1986), pp. 115-149.
- [Klin90] Kline, M. *Mathematical Thought From Ancient to Modern Times*, Volumes 1-3. Oxford University Press, New York, 1990.
- [Leve86] Leveson, N. G. Software safety: why, what, and how. *ACM Computing Surveys*, Vol. 18, No. 2 (June 1986), pp. 25-69.
- [Leve92a] Leveson, N. G. High-pressure steam engines and computer software. *Proceedings of the 14th International Conference on Software Engineering* (Melbourne, Australia, 1992), IEEE Computer Society Press, Los Alamitos, CA, 1992, pp. 2-14.
- [Leve92b] Leveson, N. G. Personal communication.
- [Li90] Li, M. and Vitányi, P. M. B. Kolmogorov complexity and its applications. *Handbook of Theoretical Computer Science*, Volume A, Elsevier, New York, 1990.
- [Litt91] Littlewood, B. and Strigini, L. Validation of ultra-high dependability for software-based systems. Preprint, 1991.
- [Loec84] Loeckx, J. and Sieber, K. *Foundations of Program Verification*. Wiley, New York, 1984.

- [Neum92] Neumann, P. G. Risks to the public. (Monthly report) *ACM Software Engineering Notes*.
- [Podg91] Podgurski, A. Reliability, sampling, and algorithmic randomness. *Proceedings of the Symposium on Testing, Analysis, and Verification (TAV4)* (Victoria, British Columbia, 1991), ACM Press, 1991, pp. 11-20.
- [Rush91] Rushby, J. and von Henke, F. Formal verification of algorithms for critical systems. *Proceedings of the ACM '91 Conference on Software for Critical Systems* (New Orleans, 1991), ACM Software Engineering Notes, Vol. 16, No. 5 (December 1991), pp. 1-15.
- [Salt89] Salter, J. H. Software performance standards under Article 2 of the Uniform Commercial Code. *Computer/Law Journal*, Vol. 9 (1989), pp. 465-489.
- [Thay78] Thayer, T. A., Lipow, M., and Nelson, E. C. *Software Reliability*, TRW Series of Software Technology 2, North Holland, New York, 1978.
- [UK88] U.K. Ministry of Defence. *Safety-Critical Computing Systems: The Procurement of Safe Computer Systems*. U.K. Interim Defence Standard 00-55, HMSO, London, 1988.
- [US89] Subcommittee on Investigations and Oversight, U.S. House of Representatives. *Bugs in the Program: Problems in Federal Government Computer Software Development and Regulation*. U.S. Government Printing Office, Washington D.C., 1989.
- [Wirs90] Wirsing, M. Algebraic Specification. *Handbook of Theoretical Computer Science*, edited by J. Van Leeuwen, MIT Press, Cambridge, MA, 1990, pp. 675-788.

A Graphical User Interface for S

Ron Baxter
Nicholas Fisher
Mark Walmsley

CSIRO Division of Mathematics
and Statistics, Sydney, Australia

Inspiration for this work comes from discussions with Mark Andrews, Rick Becker, Murray Cameron, Bill Cleveland, Nick Fisher, Kenny Leung, Branka Hoffman and Ken Yap.

1.0 The Objective

Our aim is to design and build a general purpose Graphical User Interface for S. It should give users access to the full power of S, without its complexity. Potential users can be characterised in terms of:

- Experience with other GUI's (G).
- Knowledge of data analysis and statistics (D).
- Knowledge of the S language (S).

Users who measure low in all attributes (---) require complete and detailed help and guidance, demonstrations, and an environment where it is almost impossible to unwittingly destroy data or results. At the other extreme, users who are strong in all attributes (GDS) are stress-testers, and want to be able to use their expert knowledge of S when appropriate. An important class of user is (GD-) - those users who know what they want to do, have familiarity of GUI's gained from experience with spreadsheets and WYSIWYG word processors, but who do not know the S language.

2.0 Design Considerations

Our approach is to generate a toolkit that allows menus and dialog boxes to be quickly defined for use with the S language, use this toolkit to build some fairly specific GUI's (to gain experience), then extend the toolkit to support a desktop metaphor, and finally build a general purpose GUI for S (or more specifically for S-PLUS).

Specific aims are:

- We want conformance with the Open Software Foundation's Motif GUI standard as a first stage (other versions may follow, but we consider it impractical to try and conform with several style guides simultaneously).
- We want the GUI to work with the existing unmodified S-PLUS product.
- We want to effectively use UNIX and network facilities. For example, it should be possible to run the S process on a powerful compute server and display output on the local workstation.
- We want to preserve the data analysis and statistics strengths of S, and allow users to access the power of the S language if they choose.

3.0 The Menu Building Toolkit

Specifications for the interfaces of menus and dialog boxes are written in Motif's User Interface Language (UIL). The toolkit contains a UNIX process which reads this specification when it starts up and then presents the specified menus. This process also initiates the cooperating S process and sets up communication channels between the two processes. This two process design allows the GUI process to be fully committed to listening for keyboard and mouse inputs, and showing results, while the S process evaluates expressions supplied by the GUI and returns the resulting outputs.

A user of the toolkit proceeds as follows:

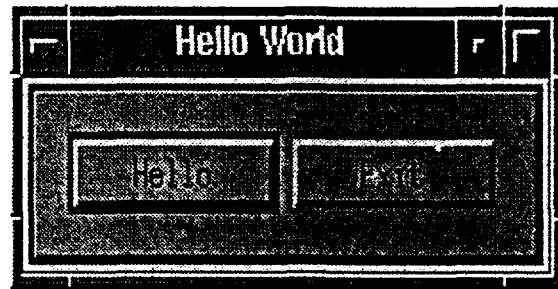
- A specification for the menus and controls is prepared using macros that simplify this process.
- Callbacks to build the S expressions for execution are embedded in this code.
- The specification is compiled using the command *Splus MC mygui.m*. This produces a compiled file *mygui.uid*.
- The compiled GUI is run with *Splus MRUN mygui*.

4.0 A Simple Example

Here is the specification for the ubiquitous *Hello World* example.

```
include(MUIS_header)
include(MUIS_IAS)
BULLETINBOARDIALOG(MainWindow, "Hello World", . . .
    MUIS_start_Splus(""), . . .
    ROWCOLUMN(. . . HORIZONTAL, 1, COLUMN, . . .
        PUSHBUTTON(, STRING, "Hello", . . . . 80, . . .
            MUIS_execute_literal_Splus_expr
                ("new.Gwin(\`Hello\`);
                vu(c(\`F ge\`, \`Hello World\`))";)
        PUSHBUTTON(, STRING, "Exit", . . . . 80, . . .
            MUIS_exit_application(), )
    )
)
```

This produces a widget with two buttons as shown.



The button on the left (Hello), when activated, pops up a graphics window with the "Hello World message" drawn by the S *vu()* function. This button can be pressed repeatedly, generating a new window each time. The other button (Exit) quits the process and removes all windows. A typical output is shown below



5.0 A Desktop for Bringing Operations and Datasets Together

The remainder of this paper deals with the concepts of the GUI which are currently in development.

- There is a strong intuitive appeal for the concept of using a desktop of icons in order to bring together datasets and operations. Manipulating the icons directly manipulates the datasets and operations. A desktop is made up of following parts:
- A menu bar with standard Motif options such as File, Edit, and Help, and statistical options, such as

Explore. These options provide access to all available operations.

- A tools panel below the Menu bar. This holds useful, or time saving, tools which may be used at any time. Possibilities include Print, Subset, Delete, Cut, Copy, Paste, Edit, and Annotate. Following the style of popular spreadsheets (Excel and Lotus), these can be small icons (16x16 rather than 32x32) so that 15-20 can be accommodated in one row. The user is given a larger set to choose from and can tailor the tool bar as required.
- A work area to display icons of S objects. Generally there is a one-for-one correspondence between S objects in a given directory (or list) and the icons displayed. However, there may be a need to create *hidden* S objects.

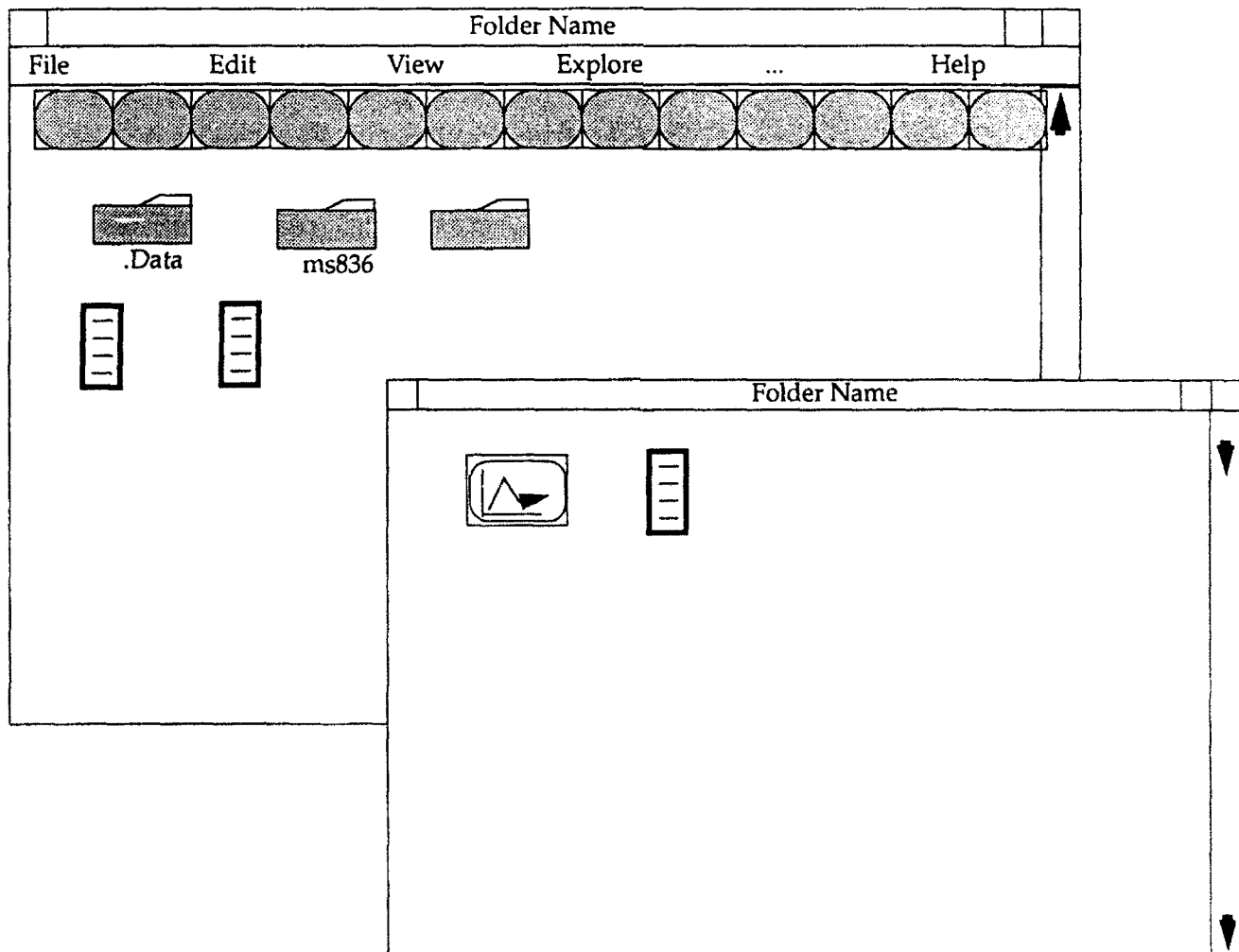
Difficulties that need to be considered include:

- Cluttered work areas where there are too many icons to conveniently deal with.
- The complexity of this model for Splus, in that many operations need a number of datasets as inputs, perhaps in a specific order.

6.0 Mouse Operations on the Desktop

We refer to the following mouse operations:

- **Select.** Clicking the left button (or a single click of a one-button mouse) highlights the icon as selected. The selected icon is the object of the action of the various tools, such as Delete, and operations.
- **Multiple Select.** Multiple items may be selected by using Shift-Select to extend selections, or by dragging a rectangle around the bunch of objects to be selected (à la Macintosh).
- **Open.** Double-clicking the left-button, or possibly single-click on another button pops up a window to display the S object associated with the icon. In the case of an operation, the displayed window is a dialog box. In the case of S data objects, the displayed



window shows the object in a format such as produced by the S `print()` or `summary()` commands. In the case of a folder, the displayed window is another desktop showing its children datasets. In the case of output objects, the displayed windows are either graphics or text windows with the appropriate results.

- **Drag.** Holding the left-button down and dragging the mouse moves an outline of the icon with the mouse. The action depends on its destination. If just within the current desktop, the icon is moved, and its new position should be remembered. If it is dragged to another desktop window, the action is to move the dataset between folders. If it is dragged to an input field of a dialog box, the action is to enter the name of the dataset into that input field. When a multiple selection is dragged to an input field the names of the members of the selection should appear in selected order. The separators between the names could be any character, such as a +, a comma, or a space, depending on context.

7.0 Look and Feel Issues

- **Icons.** Icons essentially follow the style of Microsoft Windows. The icon picture will be 32x32 in color, but with a reasonable black and white likeness (this can be seen while dragging any icon in Windows). The text field is under the icon graphic and can be any width (although very wide ones are clumsy). Selecting the icon changes the text field to a grey backdrop.
- **Object-Action Model.** The GUI works on an object-action model. The user selects an object or group of objects, and then the action to be performed on the selection. For example, to delete a dataset, select it, then select Delete from Edit menu (or the Delete tool from the toolbar). The Delete operation requests confirmation.
- **Keyboard Alternatives.** The Microsoft Windows and Motif style should be followed for these.

8.0 Datasets - how and when they will be seen

Cluttering of icons on the desktop can be controlled by providing a hierarchical structure using *folders* à la Macintosh, and UNIX directories. The user can move

collections of objects into sub-folders once a given folder is becoming unmanageable.

We see several classes of folder:

- **Normal or Desktop folder.** This contains any S objects including folders. Such a folder will probably be implemented as UNIX directories for security. Results will often be placed into a folder of this class.
- **Table folder.** This is constrained to contain vectors of equal length. Such a folder could be implemented as UNIX directories or as attached S data-frames depending on security/efficiency trade-off considerations.
- **Loop folder.** This contains an ordered set of objects which will specify a look for mass-production scenarios. These could be implemented as attached lists and could even contain expressions (as pointers to data) rather than the actual data objects. A loop folder acts differently from a normal folder when supplied as input to an operation. The ordering will be displayed by the position of icons in reading order. Where several arguments need to be supplied to the loop, each row of icons relates to one argument.

9.0 Data-frames and Lists

Data-frames and lists behave like other S objects most of the time, and can be displayed and edited, and so on. But sometimes we want to treat them like folders so that we can open them and interact with their components. The system provides two operations (commonly accessed from the tool bar) to:

- Convert a data-frame to a folder.
- Convert a folder to a data-frame (if all objects the same length), or else to a list.

This could be implemented in two ways. The folder could be created by making a UNIX directory and moving the data-frame components into this directory. Alternatively the data-frame could be attached directly without copying it into a directory. Or the implementation may involve accessing components of a data-frame or list without attaching it. The first method is safer (a system failure will not lose much data) but is slower, as copies are made (significant for large data-frames). Users, especially power S users, may like to select their preferred implementation.

10.0 Operations - How and When

- The operations offered are high-level like "Smoothing" and not specific like "loess". This means that there is not a one-to-one relationship with S functions.
- The full list of operations offered is available through a menu hierarchy. Possible names for the statistical operations include **Inspect**, **Modify**, **Compare**, **Model**, **Interpret**, **Manage Data**, **Transform** and **Display**.
- When an operation is selected from the menus, an S object of class *operation* is created in the current folder that contains the setting of the dialog box for this operation. The icon of this object appears in the folder, and is opened automatically to show the dialog box. The S object keeps a record of the partially or completely filled in state of the dialog box.
- Operations can readily be deleted at any time since they can be regenerated from the menu-bar.

11.0 Format of Dialog Boxes

- **Data fields.** Data fields can have S objects dragged to them. This results in the name of the dataset being entered in the text-field. Users may type directly to these input fields, and the knowledgeable user can enter S expressions.
- **Configuration panel.** This is a panel of widgets such as buttons, sliders, and text-fields which allow the user to configure the exact behaviour of the operation, and appearance of outputs. These may lead to other dialog boxes for fine detail.
- **Control panel.** This is a panel of buttons at the bottom of the dialog box. The buttons would be:

OK. The action specified by the dialog is carried out and the dialog box closed to an icon. An output icon (or icons) appears in an output folder (which may already exist or may be created) which then opens to reveal the main output.

Apply. The action specified by the dialog is carried out, and the dialog box remains open. An output object is created and opened.

Reset. Sets the dialog box to default values.

Cancel. Closes the dialog box to an icon.

Help. Gives context-sensitive help.

12.0 Connecting Data with Operations

Click **Open** on the required operation to reveal the dialog box. Then **drag** appropriate datasets onto the appropriate input fields. The dragging does not remove the dataset - an outline of the icon just follows the mouse to the field, and, when dropped, the visible action is the completion of the text field with the dataset name.

13.0 Output from Operations

Dialog boxes, both graph (like Histogram) and exploratory (like AOV), produce one or more output objects which may be:

- **Graphics.** For example, a Histogram dialog box should produce a histogram in the output window. There is an S object with an appropriate class corresponding to this graphics window. The graph can be recreated at any time from this object.
- **Text.** For example, an AOV dialog box should produce an AOV table, with appropriate data formatting, in the output window. This also corresponds to an S object (usually a list) with an appropriate class. There is usually a print or summary method that can generate the output window from this object.
- **Folders.** Where there is more than one output object from an operation, the objects produced are packaged into a folder which also contains an icon for the filled-in dialog box of the operation. This is as a record of the recipe which produced this output, and also can be used like any other operation. This output folder is opened as a desktop window, showing icons in a sequence, and the first output object of these is opened for viewing.

14.0 Output - New Window or Append?

By **append** we mean to add further output to the existing window - which means adding extra output to the existing S object since there is a one-for-one correspondence between S objects and windows/icons.

- **Graphics.** Appending will generally *overlay* the next graphic over the first (like contours overlaid on scatter-plots)
- **Text.** Simply append the extra text to the existing text display window.

- **Folders.** While possibly creating new graphics/text windows they should be added to the current output folder.

A common default behaviour is that, when a new operation is brought from the menu hierarchy to the desktop, a new output folder is created and contains all the plots and printouts generated from this operation in this folder. Generally the default is for each new graph and printout to generate a new window/object. The specific details of default action are decided by the designer of each operation.

To override defaults, each dialog box for an operation should show the three proposed names for output folder, graphics output, and text output. Clicking on these pops up a scrolling list selection box with alternative settings. These menus include the option of *New* as well as names of existing output objects.

15.0 Mass Production - Repeating the Same Operation

The mechanism we use is to select a group of datasets and *drag* them to an input field. This implies that the operation has to be repeated for each member of that group. *S* loops can be used to implement this, and so the simplest approach is to only allow one input field to receive a group of inputs.

The concept of a *loop folder* introduced earlier is a way of solving the problem: where groups of inputs could either mean looping or could just mean include all these variates in one use of the operation. There are tools in the tool-bar for making a normal folder become a loop folder (and *vice-versa*).

16.0 Recipes and Cakes

If all output objects are regarded as the *cakes*, then it is useful to keep a copy of the *recipe* with every *cake*. In this way, by applying some operation (possibly from the tools bar) to an output operation, it is possible to popup the appropriate dialog box with all settings and input fields completed as they were to get that output. For example, if outliers have been detected and removed, it may be appropriate to repeat a range of operations with the newly censored input file. Mass production provides a simple mechanism for such cases.

17.0 Metadata

All objects can have metadata attached using the Annotate tool. In operations, this takes the form of help (which could be edited by users to produce context-specific help, along the lines of "When we get these results, we do this..."). In datasets, metadata is seen as a form, in which the system records the creation date and time, and the user records the parent datasets, children datasets, observations on the dataset, conclusions about the dataset based on the observations, consequences of the conclusions, and general notes, such as background information on the data. This might encourage useful heuristics to be recorded. The metadata contains audit-trail information which shows this data object was derived from others.

18.0 The Tool Bar

The tool bar provides short-cuts for frequently used operations. The system provides standard tools which fill some of the bar, and these are referenced in documentation. The remaining slots can be filled by users from other suggested possibilities, or using their own functions. Some tools are:

- **Print.** Print is also found in the File menu. It prints any selected object with datasets as the result of `print()`, folders as a listing of the objects in the folder in some fairly informative format, and graphics outputs as a hardcopy of the plot.
- **Delete.** Delete is also found in the Edit menu. It follows the Microsoft Windows style of deletion.
- **Edit.** Edit is also found in the Edit menu. It applies to all objects, allowing the user to edit data, plots, and text outputs.
- **Subset.** Subset leads to a subset Dialog for the selected object. This action is only sensible for vectors, matrices and data-frames.
- **Summary.** Summary gives more concise outputs than Print. It can be usefully applied to selections of many objects.
- **Find.** Find provides a facility for finding objects by name, or class, or such in the complete data hierarchy.

Commonly, tools operate on a single selected object. They do not produce an operation object on the desktop, and so are useful for very commonly used operations where there shouldn't be a trail of actions left around.

Hence tools provide another method for avoiding clutter, and so it is important that users can choose their own.

The desktop GUI to S is an application of the toolkit which focuses on letting a statistician perform his/her job as efficiently as possible. It seeks to combine the full power of S with an intuitive GUI which makes analysing and viewing data easy. As always, though, the final test is in the use.

Adding a Survival Package to S

Terry M. Therneau, Ph.D.

Mayo Foundation
Rochester, MN 55905

I. Introduction

The August 1991 release of S, in conjunction with the book "Statistical Models in S," introduced a new set of tools for integrating the various aspects of statistical modeling into a unified *package*. Based on an object oriented paradigm, it includes methods for specifying model formulae, fitting, printed output, residuals, fitted values, and plots.

Several years ago I had created a collection of survival functions for use with S, which have subsequently been posted to the statlib archive and have been included in the S-plus package. It seemed like a good idea to try and "repackage" these using the new paradigm. The result of this effort has (mostly) lived up to expectations: it is easier to use, with greater functionality.

The focus of what follows is not a description of the survival package itself, but of the issues that arose in its design and implementation. How were the routines organized to fit into the "S model," what was particularly easy or hard, what is missing or deficient in the S tool set for this problem?

II. Date functions

Any work with medical survival data involves the subtraction or other manipulation of dates. The input data set will contain the dates of entry, follow-up, and other events (in a variety of formats). The analysis variable is usually the time from entry to a key event. A collection of date functions was created as a first "training exercise" in the new language. They form an extension to the simple julian date routine given in the S manual.

A date is defined as the number of days since 12/31/1959, i.e., 1 January 1960 is day 1. It is represented as an integer vector with class "date". The chosen baseline date is arbitrary but convenient: our group frequently moves data sets between SAS and S, and SAS dates are also based on 1/1/1960. A natural extension would be a date-time class, where the fractional portion records the time, but this has not been implemented.

A basic set of date routines was easily defined:

`is.date(x)`: true/false, is x a date

`as.date(x)`: if x is numeric, `floor(x)` if x is character, parse it otherwise fail

`mdy.date(month, day, year)`: compute the Julian date, based on a short but rather obtuse algorithm

`date.mdy(x)`: return a list giving month, day, and year

`date.ddmmyy`: formatting function

`date.mmddyy`: formatting function

So far, there is nothing very different from old S. But now we can define *methods* for special actions:

`print.date`: Since different people may have different opinions, it checks the "print.date" option. If none is set it defaults to "15Oct91" style, i.e., my preference.

`as.character.date`: The `as.character` function is called by several other S functions. For instance, `table` calls `as.category` which calls `as.character`.

`Math.date`: `atan`, `log`, `cumsum`, etc., are made illegal

`Summary.date`: `min`, `max`, and `range` result in a date all, `any`, `prod`, `sum` are illegal

`Ops.date`: `&` and `|` are illegal
`date ± numeric = date`
`date - date = numeric`
`numeric + date = date`
`*`, `/`, `^` are made illegal

Here I discovered the first fly in the ointment. Though an expression like `date/10` seems nonsensical, it arises during the axis computation when a date is used as the x or y variable of a plot. The original definitions given above had to be relaxed to allow multiplication and division, treating the date as a numeric in that case. I suspect that other functions such as `log` may also need to be relaxed as more experience is gained with these routines.

As expected from the book's example of a "factor" class, a subscript method was required that preserves the "date" class of the result: a subset of a date vector is still a date vector. Not expected were the functions that preserve the class when they should not. For example, *is.na* returns a T/F vector with class = "date". If one tries to print this result, *print.date* is invoked and the display is "1Jan60" instead of T, "31Dec59" instead of F. This was easily solved by creating a simple *is.na.date* function that drops the class:

```
is.na.date <- function(x) {
  class(x) <- NULL
  is.na(x)
}
```

A similar problem exists with *format*, but unfortunately it is not currently set up as a method in S, so that a *format.date* "override" is not possible.

The *plot.date* function reveals a shortcoming in the S inheritance model. Since method recognition depends only on the first argument, a *plot.date* function can be written which gives proper labeling to the horizontal axis when *x* is a date, but not when *y* is a date. In this case, what is really required is an *axis* method. Consideration of this along with other functions such as *mean* and *quantile* (the result should be a date-time?) leads to the conclusion that all of the basic S functions will need to be implemented as methods.

As well, some functions will need to have internal methods based "hooks" so that they can be expanded. One example is *data.frame* and its cohorts *scan*, *read.table*, and *print.data.frame*. In order to add dates as a legal class, an addition must be made internally to the function, it does not suffice to create *data.frame.date*.

III. Survival

A display of the survival routines is shown in Table 1. All told, there are about 40 S functions, 2/3 of which are "behind the scenes" and would rarely be involved directly by a user.

Some of the design issues in organizing these were:

1. Model formulae

The dependent or *y* variable in survival is usually subjected to some sort of censoring, representation of which requires an extension to the *y~x1+x2* notation of an S formula. Several ideas were considered, subject to the following S rules:

Table 1

Main routines	Class	Methods
coxph	coxph, survreg or coxphnull	print summary plot residuals predict survfit
survreg	survreg, lm	print summary plot residuals predict survfit
survdiff	survdiff	print
survfit	survfit	print summary plot lines points
survexp	survexp, survfit	print

- The left-hand side must appear to be a valid S expression for the formula to be parsed. Thus something like *stime,status~x* is illegal.
- The expression is passed to the function unevaluated and could be "torn apart" at that level. For instance, the SAS like notation *stime (status=2) ~x* could be dealt with even though *stime* is a variable and not a function; another choice would be *stime/status ~x*. Something like this is done with the vertical bar notation in the *coplot* function.
- The actual items passed forward to the model frame function must evaluate to data frames or matrices of *n* rows, or numeric vectors of length *n*: *list(stime, status)~x* won't work because the list has length 2.
- The returned model frame will have a single variable or object identified as the response.

The solution chosen was to create a "packaging" function *Surv(time,status)*. Its returned value is a matrix of 2 or 3 columns with class = "Surv" and a type attribute of "right", "left", "interval", or "counting" that identifies the censoring type. Advantages of this form are that it required the least meddling with the standard model handling routines and that we can use

Surv objects outside of the modeling language. For instance, $y \leftarrow \text{Surv}(\text{time}, \text{status})$; $\text{coxph}(y \sim x)$ is a legal construct. Methods have been defined for print, is.na, mathematical operations, and subscripting. The latter is interesting in that both single and double subscripts are allowed. If you think of Surv only as a packaging function, it is reasonable to view its result as vector of survival times—the matrix nature is only an artifact of using a matrix for the packaging medium. On the other hand, within the body of a function I often want to extract a particular component (column) of the Surv object.

Inclusion of Surv objects in a data frame remains an open issue, as it is for date objects.

2. Strata

In the Cox model there is an intercept function $\Lambda_0(t)$ rather than an intercept term in the X matrix. The X matrix should not contain a column of 1s, but dummy variables are coded as if there were a column of 1s. Multiple intercepts, referred to as *strata*, are common.

The design of S formula processing was tailor made, it seems, to handle this case. A function *strata* was created to deal with the multi-strata situation, which like *intercept*, returns a factor. A formula is processed as:

```
T ← terms (formula, specials = 'strata')
M ← model.frame (T, data)
SS ← attr (T, 'specials')$strata
T$intercept ← 0
x ← model.matrix (T[-SS], m)
```

(The actual code is slightly more complicated than this, in order to deal with SS=null or multiple strata statements.) The key is that *terms* is called with the formula, including its implicit "+1", and it is this function that makes decisions about the coding of dummy variables. After *terms* has processed the formula, the intercept is forced to be "none" and *model.matrix* is called to construct the actual X matrix.

3. Implicit interactions

The *survdiff* routine provides the survival analog of a multi group t-test. For both this routine and the Kaplan-Meier *survfit*, I decided to allow an alternate interpretation in the modeling language. That is, since the "x" variable must delineate groups, the model

```
survdiff (Surv(time, status) ~ rx + sex)
```

is processed as though the right-hand side were *interaction(rx,sex)*.

4. Returned values

Ideally, the object returned by *coxph*, *survfit*, and etc., should contain *all* of the relevant information about the fit, so that residuals and other summaries can be extracted without reference to the original data. For the simpler routines such as *survdiff* this was possible, but for *coxph* and *survreg* it was not feasible. The issue is that some of the summaries require the original X matrix (or an object of essentially the same size as X). A decision about what should be saved "by default" is a tradeoff of space for the fit object, time to reconstruct X, and an estimation of which summaries will be used most often in actual practice.

5. Inheritance

One of the more elegant features of the new system is its ability to "bootstrap" new methods onto old. The *glm* and *gam* functions inherit much of linear model's functionality by representing their results as an iteratively reweighted least squares (IRLS) solution. To the extent that survival models could do this, they also would gain.

After some exploration it became clear that this was not feasible for the Cox model, for two reasons. First, the likelihood is not represented as a sum of independent terms, straining the representation. Secondly, null models are fairly common, i.e., models with only ~1, an offset or a strata term on the right hand side. These lead to a null X matrix and the *lm* functions are not set up to handle this case.

The IRLS representation is possible for parametric survival. This was one reason for separating parametric and Cox regression into two separate functions.

IV. Missing values

The proper method for management of missing values has generated fairly intense debate within the S community. The essential issues have been that there are multiple opinions about how missing values should be handled, and that only one of them could be incorporated into the S package. Using the new S methods, we can begin to relax this second statement.

The survival routines implement a missing value strategy based on the following goals. First is that the user should be able to specify a strategy globally, as part of their setup. Second, once chosen, a missing value "plan" has impact on three areas:

1. Transformation of the input data to make it suitable for model processing.

2. The printout of the fitted model should include information about the action taken in 1.
3. Residuals and predicted values may need to be modified based on the action.

The survival routines are able to fulfill these with some simple extensions to S. Keep in mind that although the goals are clear, the following implementation of them is a trial and needs further comment and refinement.

- a. A new option "na.action" is used to signal the global default na action, e.g., `options(na.action='na.omit')`. One of the base S routines, `model.frame.default`, had to be modified to check for this.
- b. Item number 1 is already addressed by the na.action routines of S. However, in order to effect items 2 and 3, the na.action routine needs to pass along some ancillary information about what was actually done. This is added as an attribute `na.action` to the data frame. The content of the information is unrestricted, but it must have its class set appropriately, i.e., `class="omit"` for `na.omit`. The modeling routines add this to the returned fit as the `na.action` component.
- c. The `print.coxph`, `print.survdiff`, `summary.coxph`, etc., routines all make a call to `naprint(fit$na.action)`, and print out the character string that it returns.
- d. The residual and predict functions make a call to `naresid(fit$na.action, x)`, where `x` is the vector or matrix that they would have returned if there were no na.action. The routine returns a new `x`.

In particular, my own implementation of the `na.omit` function returns a vector of the deleted observation numbers. The `naprint.omit` routine returns the string "___deleted due to missing", and the `naresid.omit` routine expands `x` back to the original dimension of the input data by re-inserting the missing residuals or predicted values.

V. Summary

The new modeling and methods paradigm gives the user a large scope for innovative statistical programming. Several areas remain to be worked on, including methods for a larger number of S functions, inclusion of new data types into data frames, and extension of missing value methods, but they represent completions of the proposed framework rather than departures from it.

The Minimal Spanning Tree for Nonparametric Regression and Structure Discovery

David Banks, Department of Statistics, Carnegie Mellon University

Michael Lavine, Institute of Statistical and Decision Sciences, Duke University

Abstract: C.T. Zahn (1971) and Bhavsar and Ling (1988) describe nonparametric structure discovery methods based upon minimal spanning trees. This paper undertakes to unify and extend these ideas, resulting in an appraisal of the potential of minimal spanning trees across a range of simulated situations. We generalize minimum spanning trees to minimum spanning hypersurfaces, and discuss the value of these objects for high-dimensional data analysis.

1 Introduction

Much effort has been spent on nonparametric regression and structure discovery. Previous work has focused on smoothing techniques (Härdle, 1990), local adaptation (Cleveland and Devlin, 1988), and, to a lesser extent, robust regression (Rousseeuw and Yohai, 1984).

This project examines the use of minimum spanning tree ideas as a tool for identifying submanifolds on which probability mass concentrates. As an example, suppose one observes $(X_1, Y_1), \dots, (X_n, Y_n)$. If one is interested in nonparametric regression structure, then the usual model takes the form $Y_i = f(X_i) + \epsilon_i$ with f an unknown function. We assume that the $\epsilon_1, \dots, \epsilon_n$ are independent but not necessarily identically distributed, each with probability mode at 0. The object is to estimate f .

The proposed algorithm forms the minimum spanning tree on the data, and then prunes the tree until it contains no branches. Pruning removes the longest 10% of the edges, which typically produces a disconnected graph. Then one deletes the smallest remaining graph fragments, and trims the longest edges from the largest remaining fragment until exactly two nodes have degree 1, and all others have degree 2. This eliminates branches in the resulting graph, and is one way in which the regression perspective is enforced. Modified pruning could be used to ensure that the trimmed graph corresponds to a function; i.e., that no x value gets mapped to more than one y value.

In cases in which one is not estimating the regression function, but rather the high probability regions of a bivariate density, then one can use alternative pruning methods that permit branching and which treat the X and Y variables symmetrically. It turns out that procedures based on the minimum spanning tree are robust

to contamination of the sample by spurious values.

The next section presents graphics that illustrate the potential of the method. The third section describes a simulation study of regression structure in \mathbb{R}^2 , and compares the results of several regression strategies across a range of situations. This discovers the advantages of the spanning tree technique, and the limits of its capability. The final section uses insights gained from the simulation study to propose methodology for general submanifolds in high-dimensional spaces.

2 Illustrations

Figure 1 shows simulated data for which one wants to discover the underlying regression function. The description of the simulation that generated these are deferred to the discussion of Figure 6, so that the reader may enjoy the thrill of the hunt.

Figure 1: Raw Data

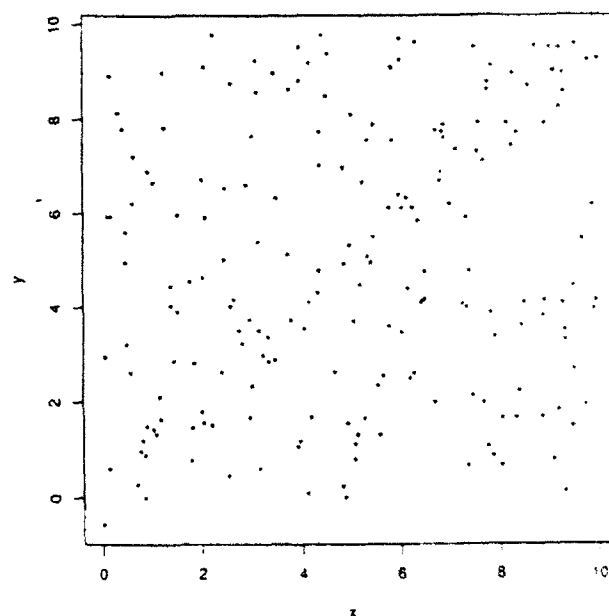


Figure 2 shows fits from three standard methods. The solid line is obtained from simple linear regression, the dotted line is obtained by quadratic regression, and the

dashed line is a kernel regression estimate (section 3).

Figure 2: Other Fits

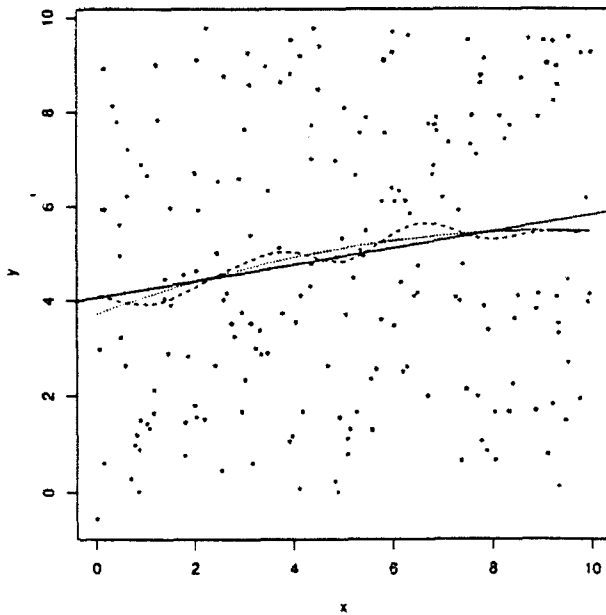


Figure 3 shows the minimum spanning tree (for Euclidean distance).

Figure 3: Spanning Tree

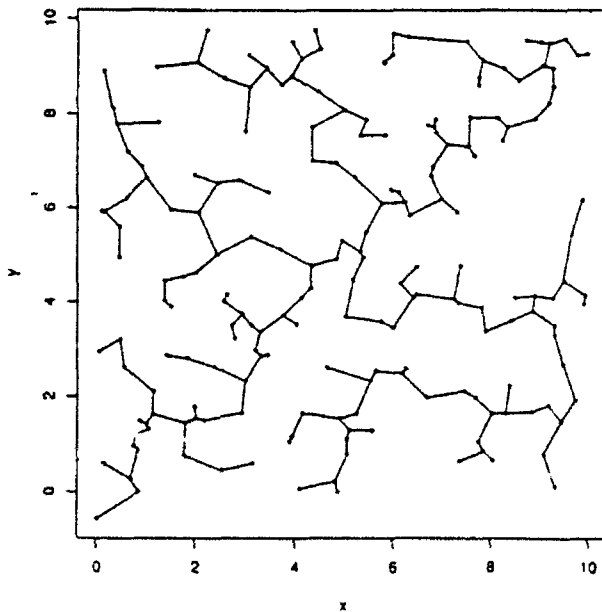


Figure 4 shows the fragments that remain after delet-

ing the 10% of the edges that have greatest length.

Figure 4: Graph Fragments

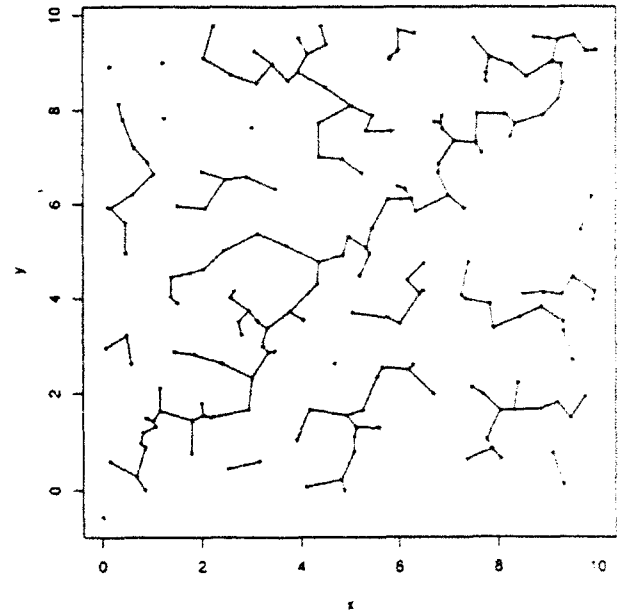
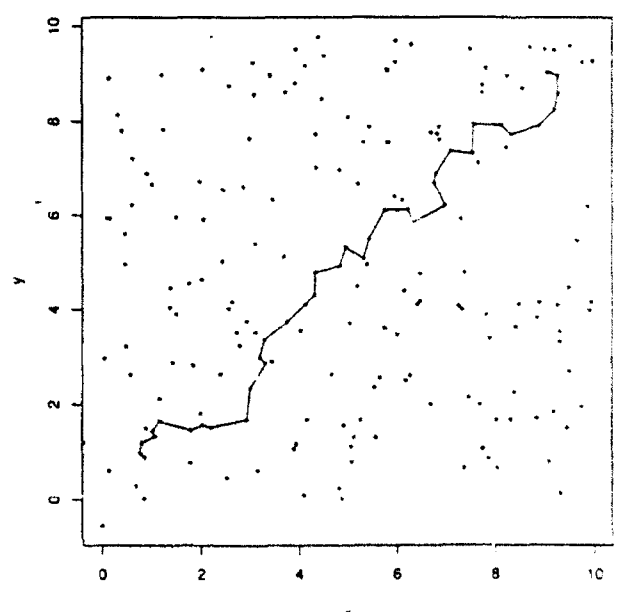


Figure 5 shows the trunk, our estimate of the regression function.

Figure 5: Trunk

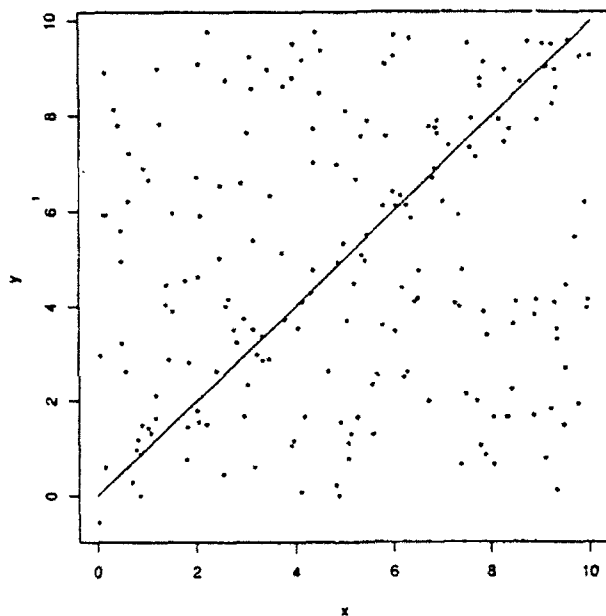


The trunk is found by trimming the longest edges from

the largest fragment in Figure 4. Trimming continues until the graph has no nodes of degree greater than 2; this precludes branching. A drawback is that some portions of the graph show one-to-many correspondence, so that the estimate is not a proper function.

Figure 6 shows the actual regression function. The points were simulated by sampling 40 points randomly from the line $y = x$ and adding $N(0, .25)$ noise. For further complication, we added 160 observations generated independently from the bivariate uniform distribution on $[0, 10] \times [0, 10]$.

Figure 6: Regression Function



The minimum spanning tree technique highlights the conditional mode of the data, whereas the competing kernel regression technique estimates the conditional mean. This enables the spanning tree technique to ignore many kinds of spurious data. Also, the spanning tree does not use smoothing, thereby avoiding the inflated bias that typically results from this operation. The minimum spanning tree relies in part upon the result from Hartigan (1981), showing that the longest gap between points is a consistent estimator of low-probability regions.

For a more complicated regression function, consider the data generated by sampling 40 points at random from the function $y = 8 + \sin x$, with $N(0, .25)$ noise and 40 spurious observations chosen independently from the bivariate uniform distribution on $[0, 10] \times [0, 10]$.

The data and function are superimposed in Figure 7.

Figure 7: Data and Regression Function

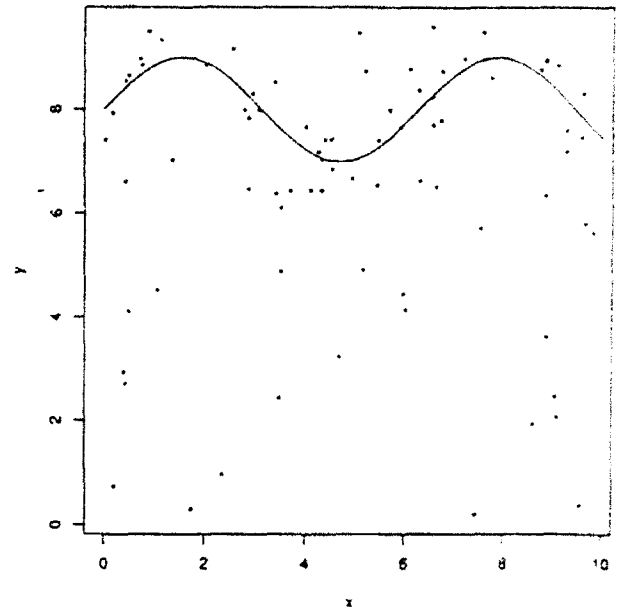


Figure 8 shows the results of the three conventional fitting procedures.

Figure 8: Other Methods

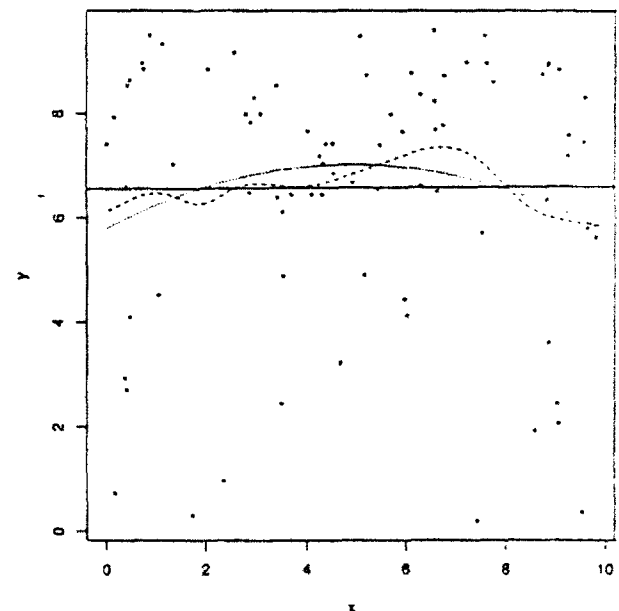
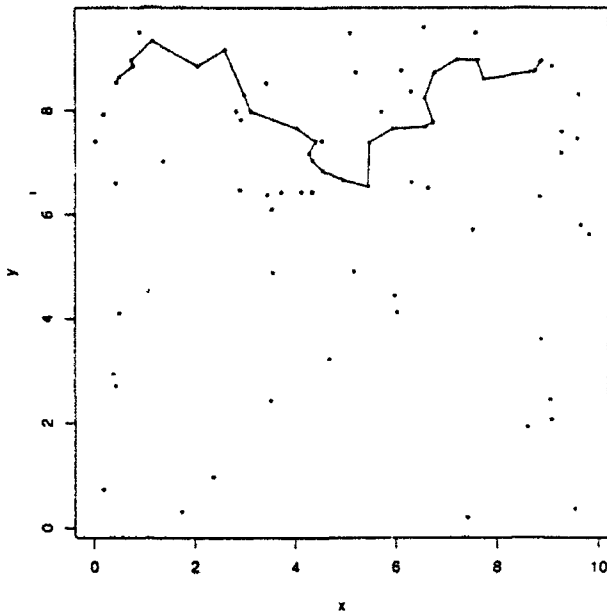


Figure 9 shows the trunk estimate of the regression function for this problem. Notice how it tracks the oscillation.

Figure 9: Trunk Estimate



3 Simulation Study

The simulation study examines 5 factors. These are:

Signal: The three regression functions are $y = x$, $y = \exp(x/4.3)$ and $y = 8 + \sin x$.

Noise: The y values are measured with normal error; the mean is zero, and the six noise levels are $\sigma = .1, .5, 1, 2, 4$ and 8 .

Sample Size: The study took n observations from the regression function, for $n = 10, 40, 60$.

Contamination: Spurious observations, sampled independently from the bivariate uniform distribution on $[0, 10] \times [0, 10]$, contaminated the data. The three levels were 0, n or $4n$ spurious values, for n the sample size.

Estimator: Five estimators were compared—linear regression, quadratic regression, kernel regression, and two methods minimum spanning tree methods.

For each combination of factor levels, 200 datasets were generated and analyzed. The competing methods are assessed in terms of integrated mean squared error. For the minimum spanning tree methods, the regression function estimator usually does not exist over the entire domain,

$[0, 10]$. For these, we inflate the estimated integrated mean squared error in proportion to the coverage, so that the results are comparable across methods. We also record the proportion of the domain for which the estimator exists.

Space limitations preclude a complete tabulation of the results. One minimum spanning tree method essentially dominated the other, and so the inferior one was dropped. Also, for the noise factor, all methods performed poorly for $\sigma \geq 4$ (with trivial exceptions, such as linear regression when $y = x$), and thus the table excludes large values of σ .

Column a indicates the total number of observations. Column b indicates the value of σ (1 implies 1, 2 implies .5 and 3 implies 1). Column c indicates the contamination (1 implies none, 2 implies that half the sample is spurious, 3 implies that 80% of the sample is spurious). Columns d, e, f and h give the estimated integrated mean squared error for linear, quadratic, spanning tree and kernel regression, respectively. Column g indicates the average percentage of the interval $[0, 10]$ for which a spanning tree regression estimate existed.

Abbreviated Simulation Results: $y = x$

a	b	c	d	e	f	g	h
10	1	1	0.02	0.06	0.07	.35	3.9
20	1	2	31.03	38.44	13.38	.54	45.0
50	1	3	55.90	57.16	34.61	.67	60.3
10	2	1	0.60	1.94	1.64	.38	5.8
20	2	2	27.51	34.61	18.26	.55	39.3
50	2	3	57.08	58.69	43.99	.65	61.8
10	3	1	2.42	5.69	6.11	.42	9.0
20	3	2	29.94	35.52	24.54	.59	40.9
50	3	3	59.18	61.24	55.30	.69	64.4
40	1	1	0.00	0.00	0.06	.10	0.5
80	1	2	21.75	22.91	2.151	.46	25.3
200	1	3	54.30	54.97	24.64	.72	56.5
40	2	1	0.12	0.19	1.36	.21	0.8
80	2	2	22.35	23.31	4.93	.53	25.8
200	2	3	53.95	54.45	22.00	.72	55.9
40	3	1	0.59	0.85	4.89	.31	1.8
80	3	2	21.75	22.68	8.44	.58	25.4
200	3	3	54.14	54.63	32.72	.71	56.0
160	1	1	0.00	0.00	0.05	.09	0.3
320	1	2	21.23	21.45	0.21	.40	23.3
800	1	3	53.36	53.48	4.78	.68	54.6
160	2	1	0.03	0.04	1.06	.23	0.3
320	2	2	21.54	21.73	1.57	.54	23.6
800	2	3	53.83	53.93	14.13	.75	55.0
160	3	1	0.11	0.19	3.63	.32	0.6
320	3	2	20.86	21.19	5.31	.58	23.1
800	3	3	53.62	53.74	19.66	.76	54.8

Abbreviated Simulation Results: $y = 8 + \sin x$

a	b	c	d	e	f	g	h
10	1	1	5.52	7.85	0.91	.35	1.7
20	1	2	33.63	37.61	13.24	.58	40.1
50	1	3	74.12	75.56	68.73	.70	77.1
10	2	1	6.14	8.34	2.63	.37	2.9
20	2	2	36.63	40.23	19.11	.63	42.5
50	2	3	72.79	74.13	54.49	.72	76.1
10	3	1	7.44	11.60	7.26	.44	7.8
20	3	2	35.37	39.55	23.14	.64	42.0
50	3	3	72.25	73.45	72.46	.71	75.2
40	1	1	4.60	3.94	0.07	.11	0.5
80	1	2	31.19	31.25	2.80	.55	30.0
200	1	3	70.24	70.26	33.28	.73	69.8
40	2	1	4.71	4.14	1.58	.25	0.8
80	2	2	31.56	31.48	10.14	.64	30.5
200	2	3	70.18	70.21	49.62	.78	69.9
40	3	1	5.10	4.72	5.77	.37	1.6
80	3	2	31.35	31.31	14.04	.64	30.4
200	3	3	70.80	70.94	56.07	.75	70.3
160	1	1	4.45	3.65	0.05	.08	0.3
320	1	2	30.31	29.84	0.43	.52	27.8
800	1	3	70.21	69.95	10.63	.75	69.1
160	2	1	4.48	3.70	1.06	.22	0.4
320	2	2	30.32	29.81	3.51	.64	27.8
800	2	3	69.66	69.40	19.95	.76	68.3
160	3	1	4.58	3.85	3.50	.34	0.6
320	3	2	30.26	29.77	9.25	.69	27.7
800	3	3	70.32	70.11	27.34	.76	69.1

Space limitations preclude the display of the results for $y = \exp(x/4.3)$.

The simulation experiment showed that for $y = x$, the minimum spanning tree method is generally superior when $\sigma \leq 1$ and contamination is present; otherwise, linear regression wins. For $y = \exp(x/4.3)$, the spanning tree method always wins when $\sigma = .1$, or when $\sigma \leq 1$ and contamination exists; otherwise, linear and/or quadratic regression do well. For $y = 8 + \sin x$, the minimum spanning tree method wins when $\sigma \leq 1$ and contamination exists, and sometimes it wins when no contamination occurs. Otherwise, kernel or linear regression win.

Regarding the coverage, note that removing the 10% of the edges that are longest sharply reduces the coverage when contamination is absent. This is because we are removing signal rather than noise. We emphasize that the minimum spanning tree method used here and described in section 1 can surely be tuned to provide better coverage and smaller integrated error.

The key contrast is between kernel regression and the minimum spanning tree method. The kernel regression smoother used is that described in Härdle (1990, p. 25)

as the Nadaraya-Watson estimator, with normal kernel and bandwidth 1. Slightly better nonparametric regression methods exist, but this was easy to program and offers a realistic benchmark for the level of performance traditional methods achieve.

4 Higher Dimensions

The encouraging performance in \mathbb{R}^2 suggests that minimum spanning tree methods may have value in higher dimensions. If one seeks to estimate hypersurface structure rather than linear structures, one must define minimum spanning surfaces in analogy to minimum spanning trees.

Our extension takes the minimum spanning $(d-1)$ -hypersurface on data in \mathbb{R}^d to be the collection of $(d-1)$ -simplices on the data that are joined at their $(d-2)$ -hyperfaces and which have the smallest possible volume. Although well defined, it is difficult to discover this surface since the spanning tree's greedy algorithm does not extend (the optimal surface can have holes, and a simple construction shows this creates problems for the analogue of the usual algorithm). However, one can generally find a very good approximation to the optimal surface, and this may suffice.

5 References

- Bhavsar, S. P. and Ling, E. N. (1988). "Are the Filaments Real?" in *The Astrophysical Journal*, Vol. 331: L63-L68.
- Cleveland, W. and Devlin, S. (1988). "Locally Weighted Regression: An Approach to Regression Analysis by Local Fitting," *Journal of the American Statistical Association*, Vol. 83:596-610.
- Härdle, W. (1990). *Applied Nonparametric Regression*, Cambridge University Press, Cambridge.
- Hartigan, J. (1981). "Consistency of Single Linkage for High Density Clusters," *Journal of the American Statistical Association*, Vol. 76: 388-394.
- Rousseeuw, P. and Yohai, V. (1984). "Robust Regression by Means of S-Estimators," in *Robust and Non-linear Time Series Analysis*, p. 256-272, ed. by Frank, Härdle and Martin, Lecture Notes in Statistics 26, Springer, NY.
- Zahn, C. T. (1971). "Graph-Theoretical Methods for Detecting and Describing Gestalt Clusters," *IEEE Transactions on Computers*, Vol. C-20, 68-86.

On Orthogonal Series Estimators for Random Design Nonparametric Regression

By Sam Efromovich *

*Department of Mathematics and Statistics, University of New Mexico
Albuquerque, New Mexico 87131*

Abstract

The problem of using orthogonal series estimators for optimal nonparametric regression is considered for the case of a random design predictors. Optimality means the best rate and constant of minimax mean integrated squared error convergence as sample size tends to infinity. The estimated regression function is assumed from the Sobolev class of functions with square-integrable α -th derivative. Both a case of given α and unknown smoothness of response function are considered; the optimal adaptive estimators are suggested. The possible extensions of the present setting, including sequential estimation and design of experiment, are considered as well.

1 Introduction

Consider the random design nonparametric regression model

$$Y_i = f(X_i) + \xi_i, \quad i = 1, 2, \dots, \quad (1)$$

where observations X_i are taken independently from a uniform distribution on the interval $[0, 1]$, and the errors ξ_i are independent normal random variables with mean 0 and variance 1, independent of $\{X_i\}$. The data set is $Z^n = (Z_1, Z_2, \dots, Z_n)$, where $Z_i = (Y_i, X_i)$. Let $\alpha = q + \kappa$, where q is nonnegative integer, and $0 < \kappa < 1$. The response function f is assumed only to belong to a subset of an α -th order Sobolev space $\mathcal{F}(\alpha, Q) = \{f : f \text{ has } q \text{ absolutely continuous and periodic derivatives, } \int_0^1 [f^2(x) + (f^{(\alpha)}(x))^2] dx < Q\}$. Here $f^{(\alpha)}$ is the Weyl α -th generalized derivative.

We are interested in the limiting Minimax Mean Integrated Square Error (MMISE):

$$\text{MMISE}(\mathcal{F}, n) = \inf_{f \in \mathcal{F}(\alpha, Q)} \sup_{f \in \mathcal{F}(\alpha, Q)} E_f \left\{ \int_0^1 (f - \hat{f}_n)^2 dx \right\}.$$

Here the inf is taken over all possible estimators \hat{f}_n based on known $\mathcal{F} = \mathcal{F}(\alpha, Q)$ and n data points $Z^n = (Z_1, Z_2, \dots, Z_n)$.

Our goal is to investigate asymptotically (when $n \rightarrow \infty$) sharp-optimal procedures for nonparametric regression, that is, the procedures with the best constant and rate of MMISE convergence, and to show that orthogonal series estimator is one of these procedures.

The problem of nonparametric estimation with optimal risk convergence has become popular in recent theoretical and applied research. It is well known that the choice of the smoothing coefficients plays a central role in an optimal nonparametric estimation. Eubank (1988) and Härdle (1990) give nice discussion of this issue.

In our problem the optimal smoothing coefficients are functions of α and Q which are unknown for most practical problems. Hence a key point for optimal nonparametric regression is an estimation for the case of unknown α and Q . Efromovich (1986) suggested a sharp-optimal adaptive estimator for an orthogonal series estimator and equidistant predictors. Speckman (1985) solved the similar problem for the case of spline estimators and equidistant predictors. The random design setting is considered in this paper.

The most relevant publications on nonparametric regression are those by Nussbaum (1985), Speckman (1985), Golubev and Nussbaum (1990). These authors establish sharp-optimal convergence of MMISE for the case of a fixed sample size n and a Sobolev class of functions over equidistant points $\{x_i\}$. The suggested method of estimation is spline smoothing.

Section 2 is restricted to the case of the model (1) and for this model a sharp-optimal procedure is suggested. An adaptive estimation, when α and Q are unknown, is considered in section 3. Extensions of the main setting are discussed in section 4 including sequential approach and bona fide estimation. Many examples, including missing value situations, applications to grouped, censored or truncated data, finite mixtures models, are the particular cases of these extensions.

*This research supported by NSF grant

2 Procedure of a sharp-optimal orthogonal series estimation

The main goal of this section is to find for the model (1) an optimal estimator for a response function. This problem is converting into finding a sharp-optimal lower bound for MMISE and a sharp-optimal estimator.

The following result gives us a lower bound for MMISE convergence. Hereafter $o(1) \rightarrow 0$ as $n \rightarrow \infty$.

Theorem 1. Let α, Q , and n are given. Then

$$\inf_{f \in \mathcal{F}(\alpha, Q)} \sup_{f \in \mathcal{F}(\alpha, Q)} E_f \left\{ \int_0^1 (f(x) - \hat{f}_n(x, \alpha, Q, Z^n))^2 dx \right\} \geq P n^{-2\alpha/(2\alpha+1)} (1 + o(1)), \quad (2)$$

where \inf is over all possible estimators $\hat{f}_n(x, \alpha, Q, Z^n)$ and

$$P = Q^{1/(2\alpha+1)} (2\alpha/(2\pi(\alpha+1)))^{2\alpha/(2\alpha+1)} (2\alpha+1)^{1/(2\alpha+1)}.$$

The best possible convergence (2) of minimax risk gives an orientation for construction an estimator. It is useful to note that this best convergence is the same as for the case of the equidistant points $\{x_i\}$ (see Nussbaum (1985)).

For proving that (2) gives a sharp lower bound we are investigating two different orthogonal series estimates. The first estimate is only rate-optimal, but this estimator is very simple and will be used as a pilot estimator for construction a sharp-optimal one. This estimator is a projection estimator. The second estimator is a smoothing orthogonal series estimator, and smoothing coefficients are prior fixed for the known α and Q and are estimated for unknown α and Q .

The underlying idea is based on the following well known representation of the Sobolev class of functions. Using the classical Fourier trigonometric basis $\{\varphi_j\}$, where $\varphi_0(x) = 1$, $\varphi_{2i-1}(x) = \sqrt{2} \sin(2\pi i x)$, and $\varphi_{2i}(x) = \sqrt{2} \cos(2\pi i x)$, $i = 1, 2, \dots$, we can rewrite $\mathcal{F}(\alpha, Q)$ as $\mathcal{F}(\alpha, Q) = \{f : f(x) = \sum_{i=0}^{\infty} \theta_i \varphi_i(x); \theta_0^2 + \sum_{i=1}^{\infty} [1 + (2\pi i)^{2\alpha}] [\theta_{2i-1}^2 + \theta_{2i}^2] \leq Q; \theta_i = \langle f, \varphi_i \rangle\}$, where the inner product $\langle f, \varphi_i \rangle = \int_0^1 f(x) \varphi_i(x) dx$.

From the last definition it is natural that a projection estimator

$$\hat{p}(x, J, n_1, n_2) = \sum_{j=0}^J \hat{p}_j(n_1, n_2) \varphi_j(x) \quad (3)$$

is rate-optimal for some J , n_1 , and n_2 , where

$$\hat{p}_j(n_1, n_2) = (n_2 + 1 - n_1)^{-1} \sum_{l=n_1}^{n_2} Y_l \varphi_j(X_l) \quad (4)$$

is the method of moments estimate of θ_j .

The following theorem gives us a more precise result. We use the notation $[x]$ for the integer part of x .

Theorem 2. The projection estimator $\hat{p}(x, J, 1, n)$ with $J = [n^{1/(2\alpha+1)}]$ is an asymptotically rate-optimal and

$$\sup_{f \in \mathcal{F}(\alpha, Q)} E_f \left\{ \int_0^1 (f(x) - \hat{p}(x, J, 1, n))^2 dx \right\} < n^{-2\alpha/(2\alpha+1)} (1 + 2Q) (1 + 2n^{-1/(2\alpha+1)}) \quad (5)$$

Note that this very simple projection estimator with only one smoothing parameter J (window size) gives us a rate-optimal risk convergence and simultaneously a strict (nonasymptotic) upper bound (5) for risk convergence.

We are now ready to define a sharp-optimal estimator which has both optimal rate and optimal constant of MMISE convergence. We again use an orthogonal series estimator but with a different estimator for θ_j and with special fixed smoothing coefficients.

The suggested linear estimator is

$$\hat{f}(x, N, J, r+1, n) = \hat{p}(x, J, 1, n) + \sum_{j=J/2+1}^N (1 - (j/N)^\alpha) [\hat{\theta}_{2j-1} \varphi_{2j-1}(x) + \hat{\theta}_{2j} \varphi_{2j}(x)], \quad (6)$$

where $\hat{\theta}_j = \hat{\theta}_j(J, r+1, n)$,

$$\hat{\theta}_j(J, r+1, n) = (n-r)^{-1} \sum_{l=r+1}^n [Y_l - \hat{p}(X_l, J, 1, r)] \varphi_j(X_l)$$

is a modified method of moments estimator of θ_j , and

$$N = N(n, \alpha, Q) = \lfloor [n(2\alpha+1)(\alpha+1) \times Q / (2\alpha(2\pi)^{2\alpha})]^{1/(2\alpha+1)} \rfloor + 2 \lfloor \ln(n) + 2 \rfloor$$

Hereafter $r = r(n) = \lfloor n / \ln(n+1) \rfloor + 1$. The statistics and sequences defined up to this point can be chosen in many different ways.

Theorem 3. Let $n > 1$ and $J = 2 \lfloor \ln(n) + 1 \rfloor$. Then the linear estimator $\hat{f}(x, N, J, r+1, n)$ defined in (6) is asymptotically minimax and

$$\sup_{f \in \mathcal{F}(\alpha, Q)} E_f \left\{ \int_0^1 (f(x) - \hat{f}(x, N, J, r+1, n))^2 dx \right\} = P n^{-2\alpha/(2\alpha+1)} (1 + o(1)).$$

3 Adaptive estimator

For the case of unknown α and Q the method of smoothing is based on the estimation the optimal smoothing coefficients. Introduce some new notation. Let $d(0) = 2[\ln(n) + 1]$, $d(k) = d(k-1) + 2k$, $T(k) = \{d(k-1) + 1, d(k-1) + 2, \dots, d(k)\}$, $J(0) = d(0)$, $J(k) = [d(k-1)/(1 + \ln^2 k)]$, where $k = 1, 2, \dots$. Recall that $r = r(n) = [n/\ln(n+1)] + 1$. Define statistics

$$\hat{\Theta}(k, n) = (2k)^{-1} \sum_{j \in T(k)} [\hat{\theta}_j(J(k), r+1, n)]^2 - (n-r)^{-1},$$

$$\hat{L}(k, n) = \frac{\hat{\Theta}(k, n) \chi(\hat{\Theta}(k, n) - \ln^{-1}(k+1)(n-r)^{-1})}{\hat{\Theta}(k, n) + (n-r)^{-1}},$$

where $\chi(x) = 1$ if $x > 0$ and $\chi(x) = 0$ if $x \leq 0$.

Theorem 4. Assume that α and Q are unknown but fixed and $\alpha > 1/2$. Then an adaptive estimator

$$\begin{aligned} \hat{f}_a(x, n, S) &= \hat{p}(x, J(0), 1, n) \\ &+ \sum_{k=1}^S \hat{L}(k, n) \sum_{j \in T(k)} \hat{\theta}_j(J(k), r+1, n) \varphi_j(x) \end{aligned} \quad (7)$$

is asymptotically minimax and

$$\begin{aligned} \sup_{f \in \mathcal{F}(\alpha, Q)} E_f \left\{ \int_0^1 (f(x) - \hat{f}_a(x, n, S))^2 dx \right\} \\ = P n^{-2\alpha/(2\alpha+1)} (1 + o(1)), \end{aligned}$$

where $S = [n^{1/4}]$.

The suggested adaptive estimator is asymptotically efficient over all linear estimators in the sense of Efromovich (1985).

4 Extensions

Sequential estimation. Let consider the wider class of estimators which allow the sample size τ to be statistic (stopping time) with a moment restriction $\sup E_f \{(\tau/n)^\beta\} \leq 1$, where \sup is over $f \in \mathcal{F}(\alpha, Q)$ and β is a fixed constant not less than one.

Efromovich and Pinsker (1991) show that for the case of given α and Q sequential procedures can not improve the convergence of MMISE. The same conclusion is true if α and Q are unknown but fixed and $\alpha > 1/2$. However, the optimal estimators are surely different for these two cases.

More precisely if α and Q are given, then sharp-optimal sequential estimator is the linear estimator (6)

with the fixed sample size $\tau = n$. If α and Q are unknown but fixed and $\alpha > 1/2$ then sharp-optimal sequential estimator is the adaptive estimator (7) with the fixed sample size $\tau = n$.

Sequential estimation with guaranteed precision. The problem is a sequential minimax estimation with risk no greater than ϵ . Subject to this condition, optimality of a sequential procedure is defined as a minimization of the maximum over $f \in \mathcal{F}$ of mean stopping time, that is an optimal procedure is a sequential plan with a minimax stopping time subject to guaranteed precision of estimation.

We are interesting in the sharp limit (as $\epsilon \rightarrow 0$) of the Minimax Mean Stopping Time $MMST(\mathcal{F}, \epsilon) = \inf \sup E_f \{\tau\}$. Here the \sup is over $f \in \mathcal{F}$ and the \inf is over all possible sequential plans $\Pi_\epsilon = (\{f_m(x), m = 1, 2, \dots\}, \tau)$ based on known \mathcal{F} and such that the Mean Integrated Squared Error (MISE) is not greater than ϵ for any $f \in \mathcal{F}$, that is, that

$$\sup_{f \in \mathcal{F}} E_f \left\{ \int (f_\tau(x) - f(x))^2 dx \right\} \leq \epsilon \quad (8)$$

The following lower bound for MMST is valid

$$MMST(\mathcal{F}(\alpha, Q), \epsilon) \geq n^*(\alpha, Q, \epsilon) (1 + o(1)) \quad (9)$$

as $\epsilon \rightarrow 0$, where

$$n^*(\alpha, Q, \epsilon) = [(\epsilon P^{-1})^{-(2\alpha+1)/2\alpha}] + 1. \quad (10)$$

For given α and Q a plan with a prior fixed sample size n is optimal. The case of unknown α and Q is significantly more complex problem and in this case a sequential procedure is optimal. An idea of this sequential estimation is the following.

It is assumed that α and Q are fixed and that for given γ the inequalities $1/2 \leq \gamma < \alpha \leq 2\gamma < \infty$ are valid. If α , Q , and $\tau = n$ are given then in accordance with Efromovich (1986) the MISE of a minimax estimator is asymptotically equal to $(1 + o(1))R_n$, where

$$R_n = n^{-1} \sum_{k=1}^{S'} (2k) \frac{\Theta_k}{\Theta_k + n^{-1}} + \sum_{k=S'+1}^{S''} (2k) \Theta_k, \quad (11)$$

$\Theta_k = (2k)^{-1} \sum_{j \in T_k} \theta_j^2$, $S' > [(2N)^{1/2}] + 1$, and S'' is any sequence of natural numbers such that $S' < S''$ and $N = o(1)(S'')^2$ as $n \rightarrow \infty$. Efromovich (1985) proved the following asymptotical equality

$$\sup_{f \in \mathcal{F}(\alpha, Q)} R_n = P n^{-2\alpha/(2\alpha+1)} (1 + o(1)) \quad (12)$$

The equalities (11) and (12) give us an idea of designing a stopping time. We do not know $f(x)$ and consequently Θ_k , nevertheless it is possible to estimate Θ_k

fairly well, use these estimates for estimation R_n , and then find an optimal stopping time.

An interesting feature of this problem is that the condition $\gamma < \alpha \leq 2\gamma$ is necessary for the sharp-optimality the lower bound for MMST. This discussion may be found in Efremovich and Pinsker (1991).

General setting for iid observations. This is a setting when we observe $Z^n = (Z_1, Z_2, \dots, Z_n)$ with the $Z_i = (Y_i, X_i)$ iid according to distribution with density $p(y, x) = \pi(x)p(y|f(x))$. More exactly we consider a statistical experiment $E_{Y|\theta}^n = \{\mathcal{A}^n, \mathcal{U}^n, \mu^n, P_\theta^n, \theta \in \Theta\}$. Hereafter $E_{Y|\theta}^n$ is a product of n identical statistical experiments $E_{Y|\theta} = \{\mathcal{A}, \mathcal{U}, \mu, P_\theta, \theta \in \Theta\}$, where μ is a σ -finite measure in \mathcal{U} and all probability measures $P_\theta, \theta \in \Theta$ are absolutely continuous with respect to μ and $p(y|\theta) = dP_\theta/d\mu$ is a conditional density.

Let the following Hajek conditions of regularity, which are sufficient for uniform local asymptotic normality of this experiment, be true:

R1. For every $\theta \in \Theta$ the condition densities $p(y|\theta)$ are absolutely continuous in θ for all $y \in \mathcal{A}$ and $\theta \in \Theta$.

R2. For every θ derivative $p'(y|\theta) = \partial p(y|\theta)/\partial \theta$ exists for μ -almost all $y \in \mathcal{A}$.

R3. The Fisher information

$$I(\theta) = \int_{\{y: p(y|\theta) > 0\}} \left[(p'(y|\theta))^2 / p(y|\theta) \right] \mu(dy)$$

exists, is continuous, and is bounded below from zero.

Define a localized Ellipsoid $\mathcal{F}(f_0, \rho, \alpha, Q) = \{f : \int_0^1 (f_0(x) - f(x))^2 dx \leq \rho, f(x) - f_0(x) \in \mathcal{F}(\alpha, Q)\}$.

If, for example, function $\pi(x)I(f_0(x))$ is continuous in x and bounded below from zero over an interval $[0, 1]$ then

$$\inf_{f \in \mathcal{F}(f_0, \rho, \alpha, Q)} \sup_{f \in \mathcal{F}(f_0, \rho, \alpha, Q)} E_f \left\{ \int_0^1 (\hat{f}_n(x) - f(x))^2 dx \right\} \geq P(nF)^{-2\alpha/(2\alpha+1)} (1 + o(1)) \quad (13)$$

where $o(1) \rightarrow 0$ as first $n \rightarrow \infty$ and then $\rho \rightarrow 0$,

$$F = 1 / \int_0^1 [\pi(x)I(f_0(x))]^{-1} dx \quad (14)$$

and inf is over all estimators \hat{f}_n .

Hence F is a new factor for a nonparametric Fisher information quantity when we consider this general setting, that is, when the distribution of the predictors is not necessary uniform, family of conditional distribution is not necessary normal location one, and the minmax is considered around a given function f_0 which is not necessary equal to zero.

A method of a sharp-optimal estimation is an analog to the scoring estimator with $o(n^{-1/2})$ -convergent orthogonal series pilot estimator. Let $\tilde{f}_m(x) = \tilde{f}(x, Z^m)$ be a $o(n^{-1/2})$ -convergent estimator, that is, that

$$\sup_{f \in \mathcal{F}(\alpha, Q)} E_f \left\{ \int_0^1 (\tilde{f}_m(x) - f(x))^2 dx \right\} = o(1)n^{-1/2}$$

and $m = m(n) = o(1)n$. Then, under some additional conditions of regularity, a nonparametric scoring estimator

$$\begin{aligned} \hat{f}_n(x) = & [\tilde{f}_m(x)]_N + (n-m)^{-1} \sum_{l=m+1}^n [\pi(X_l)I(\tilde{f}_m(X_l))]^{-1} \\ & \times (p'(Y_l|\tilde{f}_m(X_l))/p(Y_l|\tilde{f}_m(X_l))) \sum_{j=0}^N (1 - (j/N)^\alpha) \\ & \times [\varphi_{2j-1}(X_l)\varphi_{2j-1}(x) + \varphi_{2j}(X_l)\varphi_{2j}(x)] \end{aligned} \quad (15)$$

is sharp-optimal. Here $[f(x)]_N = \sum_{j=0}^{2N} f_j \varphi_j(x) < f, f_j > \times \varphi_j(x)$.

Hence in general setting the problem of the sharp-optimal estimation is converted into a $o(n^{-1/2})$ -convergent estimation. If $\alpha > 1/2$ then this pilot estimator exists and may be constructed by using orthogonal series estimator.

An applications of this method are different applied problems, including location and scale distribution families, mixtures, as well as missing data situations and applications to grouped, censored, or truncated data.

As an example consider the case when indirect observations T 's are censored, say at fixed point a . The direct observations Y 's can then be represented as $Y_l = T_l$ if $T_l < a$ and $Y_l = a$ otherwise. Hence the value a of Y has no significance. Suppose that $g(t|f(x))$ is a conditional density of T given $f(X)$. Then the direct observations (Y, X) 's are iid with density $\pi(x)p(y|f(x))$ where $p(y|\theta) = g(y|\theta)$ if $y < a$ and $p(y|\theta) = 1 - \int_{-\infty}^a g(t|\theta)\mu(dt)$ if $y = a$ with respect to the measure ν which is equal to measure μ on $(-\infty, a)$ and assigns measure 1 to the point $y = a$.

Hence this censored nonparametric regression is a particular example of the considered general setting. The suggested nonparametric scoring method gives us a sharp-optimal estimation, a pilot estimator is a traditional for this problem general method of moments estimator.

Design of experiment. The investigated sharp-optimal risk convergence is a functional of $\pi(x)$. Suppose that $f_0(x)$ is given. Then it is of interest to minimize this convergence further by optimal design of $\pi(x)$. By

Cauchy-Schwartz inequality

$$\int_0^1 [\pi(x)I(f_0(x))]^{-1} dx \geq \left(\int_0^1 I^{-1/2}(f_0(x)) dx \right)^2$$

with equality for

$$\pi^*(x) = I^{-1/2}(f_0(x)) / \int_0^1 I^{-1/2}(f_0(t)) dt. \quad (16)$$

The design of experiment with this density is optimal.

The response function f_0 is prior unknown, but for some settings optimal design does not depend on f_0 . For example in traditional setting $Y_i = f(X_i) + \xi_i$ the Fisher information quantity is equal to constant and therefore the optimal design is the uniform distribution regardless on unknown response function. For general setting instead of unknown $f_0(x)$ a pilot estimator may be used.

Unknown distributions. An important problem is a sharp-optimal estimation for the case of unknown $\pi(x)$ and $p(y|\theta)$. For estimation these densities an adaptive orthogonal estimator is suggested in Efromovich (1985). Adaptive scoring methods are well known, Bickel *et al.* (1992) can serve as a good reference.

Nonrandom design. Nonrandom design is treated in the same way. For this setting instead of random predictors with density $\pi(x)$ the predictors are points x_{in} which are generated by pseudo-density $\pi(x)$ on $[0, 1]$ such that $\int_0^{x_{in}} \pi(x) dx = i/n$. All conclusions for the random design regression are expended on this nonrandom design.

Bona fide estimator. It is typical for different applications that a response function belongs to a given class of functions, such as nonnegative functions or integrated to given constant functions. This problem as well as an estimation of nonperiodic function may be treated similarly to Efromovich and Marron (1991). The method developed by Eubank and Speckman (1990) for rate-optimal estimation of nonperiodic functions can be also applied for sharp-optimal estimation. Nussbaum (1985) shows that smoothing spline estimators are sharp-optimal for nonperiodic response functions as well.

5 Conclusion

The proposed orthogonal series estimator has the sharp-optimal property of MMISE convergence when sample size tends to infinity. This estimator is sharp-optimal both for the case of known and unknown smoothness of the estimated response function.

An interesting feature of this estimator is that it is also optimal among sequential estimators with a restricted

moment of a stopping time. A special sequential procedure is necessary only for sharp-optimal estimation with guaranteed precision and unknown smoothness of the estimated response function.

This optimal feature of the orthogonal estimator is valid for the general setting as well. The general setting includes important applied problems such as missing data, grouped and censored data, different location and scale families of distributions.

6 References

- Bickel, P.J., Klaassen, C.A.J., Ritov, Y. and Wellner, J.A. (1992) *Efficient and Adaptive Estimation in Semiparametric Models*. Forthcoming Monograph. John Hopkins Univ. Press, Baltimore.
- Efromovich, S. (1985). Nonparametric estimation of a density with unknown smoothness. *Theory of Probab. Applications* 30 557-568.
- Efromovich, S. (1986). The adaptive algorithm of nonparametric regression. In *Abstracts of Second IFAC Symposium on Stochastic Control*. Vilnius 2 112 - 114.
- Efromovich, S. (1989). On sequential nonparametric estimation of a density. *Theory of Probab. Applications* 34 228-239.
- Efromovich, S. and Marron J.S. (1991). Orthogonal series estimation of aperiodic densities. *Technical report*, Stat. Dept. Univ. Conn., Storrs.
- Efromovich, S. and Pinsker, M.S. (1991). On sharp-optimal estimation for random design nonparametric regression. *Technical report*, Stat. Dept. Univ. Conn., Storrs.
- Eubank, R.L. (1988). *Spline smoothing and nonparametric regression*. New York: Dekker.
- Eubank, R.L. and Speckman, P. (1990). Curve fitting by polynomial-trigonometric regression. *Biometrika*, 77, 815 - 827.
- Golubev, G.K. and Nussbaum, M. (1990). A risk bound in Sobolev class regression. *Ann. Statist.* 18 758-778.
- Härdle, W. (1990). *Applied nonparametric regression*. Cambridge University Press.
- Nussbaum, M. (1985). Spline smoothing in regression models and asymptotic efficiency in L_2 . *Ann. Statist.* 13 984-997.
- Speckman, P. (1985). Spline smoothing and optimal rates of convergence in nonparametric regression models. *Ann. Statist.* 13 970-983.

A LARGE SCALE GAUSSIAN SMOOTHING ALGORITHM

R. Kent Goodrich¹, Ranjit M. Passi² and Mark Limber³¹ Department of Mathematics, University of Colorado, Boulder CO 80309² Institute for Naval Oceanography, Stennis Space Center, MS 39529³ University of Waterloo, Waterloo Ontario, Canada N2L 3G1

ABSTRACT

In many practical situations, data constrained to a uniformly spaced, large dimensioned orthogonal grid in two dimensional space is to be smoothed using a convolution with a Gaussian weighting function, $w(r) = \exp(-r^2/b^2)$. The smoothing operation may have to be performed repeatedly. Such situation occurs quite frequently in atmospheric and oceanic data assimilation, where model observations are optimally combined with the observed data. Because of the large numbers of data points, direct application of this smoothing operation may be computationally prohibitive. We derive a computationally efficient algorithm to estimate this smoothing operator by employing polynomial operators, $P(D_i)$, on the gridded data, $f(m, n)$, where D_1 and D_2 are averaging operators in the X and Y directions: $D_1 f(m, n) = [f(m+1, n) + f(m-1, n)]/2$; $D_2 f(m, n) = [f(m, n+1) + f(m, n-1)]/2$. The polynomial P is the interpolating polynomials over the expanded Chebychev points, x_i on the interval $[a, b]$, i.e.,

$$x_i = \frac{1}{2} \left[a + b + (a - b) \cos \left(\frac{2i+1}{2n+2} \pi \right) / \cos \left(\frac{\pi}{2n+2} \right) \right]$$

of the function $g(\theta) = 1 + \sum_{s>0} e^{-s^2/b^2} (2 \cos 2\pi s \theta)$. Using spectral analysis, the smoothing achieved through matrix multiplication is shown to be well approximated by the operation $P(D_2)P(D_1)f(m, n)$. This estimate is nearly optimal in the infinity norm. Simulations show that the degree of the operator polynomials required for satisfactory approximation is quite small, which makes the algorithm computationally efficient by a factor of ~ 25 over the actual matrix multiplication procedure. The algorithm is being generalized to higher dimensional space, and with ellipsoidal (instead of spherical) contours of the weighting function.

1. INTRODUCTION

This paper describes a computationally efficient algorithm of smoothing a data field constrained to a uniformly spaced, large dimensioned orthogonal grid in a two dimensional space using a Gaussian weighting function, $w(r) =$

$\exp(-r^2/b^2)$. Such a smoothing operation is often performed in meteorology and oceanography. In our application, the problem arises while iteratively minimizing a quadratic functional using a conjugate gradient algorithm where it is required to multiply a covariance matrix, Σ_m , with a vector, g , defined on the grid. With such multiplications occurring repeatedly, and with the dimensions being extremely large, there is a need to find a computationally efficient algorithm to estimate this multiplication.

Derber and Rosati (1989) implemented an efficient algorithm to compute $\Sigma_m g$, when Σ_m is based on a Gaussian function. The algorithm looks at the continuous analog of the matrix multiplication by expanding $g(x, y)$, the continuous analog of g , in a Fourier expansion, and approximates Gaussian smoothing by applications of a large number of Laplacian operations. Although, this algorithm is computationally much more efficient than the actual matrix multiplication, it still needs further enhancement for it to be used on an operational basis. The enhancement that we propose is based on the direction suggested by the Derber-Rosati algorithm of approaching the problem in the Fourier domain.

This new technique uses spectral decomposition in discrete coordinates. First, using the convolution property of the Fourier transforms, the matrix multiplication $\Sigma_m g$ is expressed as an inverse of the product of Fourier transforms of two two-dimensional arrays, $g(m, n)$ and $\sigma(k, l)$, where $\sigma(k, l)$ are based on the Gaussian function. This expression of $\Sigma_m g$ is then shown to be approximated by the one obtained by the application of polynomials in simple averaging operators on the vector g .

First, in Section 2, we develop the statistical formulation in terms of data assimilation application leading to the need for an efficient algorithm for matrix multiplication. In Section 3, by a brief description of the Gaussian smoothing algorithm (Derber 1990, personal communication) used by Derber and Rosati (1989). In Section 4 we develop our new smoothing algorithm, which is based on operators that are polynomials in simple averaging operators. We then compare the Gaussian smoothing achieved by the two algorithms with that obtained by the actual ma-

trix multiplication. In concluding remarks, we mention the future improvements and enhancements of this algorithm.

2. STATEMENT OF THE PROBLEM

In meteorology and oceanography, scientists often combine the observational data with the output of a numerical general circulation model to arrive at the best current state of the atmosphere/ocean. The process of model/data combination, referred to as data assimilation, is performed using the method of least squares to achieve minimum variance. The statistical framework is as follows.

We want to estimate the true state of the ocean, Θ , by combining the model output, T_m , and observations T_o . The vectors Θ and T_m are $N \times 1$ vectors defined on the model grid, G_m ; the observation vector, T_o is an $M \times 1$ vector defined on the observation grid G_o . Usually, $M \ll N$, and T_o can not provide an adequate representation of Θ . The assimilation has to be performed on the G_m , as the resultant estimate will be used as initial conditions for numerical integration of the model equations.

We assume there exists a mapping such that $D(G_m) = G_o$. Then Θ_o , the true state of the ocean at the observation grid, can be written as $\Theta_o = D(\Theta)$. Often D is assumed to be a linear mapping so that:

$$\Theta_o = D\Theta \quad (1)$$

Assuming the model output T_m and the the observations T_o as unbiased, we can write the following linear model:

$$\begin{pmatrix} T_m \\ T_o \end{pmatrix} = \begin{pmatrix} \Theta \\ D\Theta \end{pmatrix} + \begin{pmatrix} e_m \\ e_o \end{pmatrix} \quad (2)$$

where e_m and e_o are the zero mean random model and observation error vectors, respectively. Let Σ_m be the covariance matrix of e_m and Σ_o be the covariance matrix of e_o . Assuming the errors in T_o and T_m as statistically independent, the least squares solution to the true state Θ is obtained by minimizing the quadratic functional:

$$Q = (T_m - \Theta)' \Sigma_m^{-1} (T_m - \Theta) + (T_o - D\Theta)' \Sigma_o^{-1} (T_o - D\Theta) \quad (3)$$

where the prime indicate matrix transposition. It is customary to write Q in terms of corrections to the model values:

$$Q = T_c' \Sigma_m^{-1} T_c + (DT_c - T_{co})' \Sigma_o^{-1} (DT_c - T_{co}), \quad (4)$$

where $T_c = T_m - \Theta$ and $T_{co} = T_o - DT_m$.

Ordinarily, this minimization should be quite simple, but for the fact that the dimension N of the model grid, and hence that of the covariance matrix, Σ_m , are extremely large so that routinely used procedures are quite inadequate and efficient minimization algorithms must be devised.

The data assimilation algorithm is determined by specifying the covariance matrices Σ_m and Σ_o , and by the method of minimizing Q . With a given Σ_m , one efficient algorithm for minimizing Q is the preconditioned conjugate gradient algorithm which avoids computing Σ_m^{-1} (Navon and Legler, 1987). It iteratively reduces the gradient vector g to zero; i.e.,

$$\Sigma_m^{-1} T_c + D' \Sigma_o^{-1} (D(T_c) - T_{co}) = 0. \quad (5)$$

Each iterative step computes

$$h = \Sigma_m g. \quad (6)$$

Unless the Σ_m is diagonal, the matrix multiplication in (6) is quite expensive. This is especially so for repeated applications. The situation becomes more ponderous when, due to large dimensions, it may not be possible to store Σ_m in computer memory, and Σ_m has to be computed again and again.

We present an algorithm that alleviates the above difficulties, when the covariance structure of Σ_m is based on a Gaussian function

$$C(r) = a \exp(-r^2/b^2). \quad (7)$$

and g is constrained to a uniformly spaced, large dimensioned orthogonal grid in a two dimensional space, computationally efficient algorithms can be devised that are quite accurate, and a lot faster than the actual matrix multiplication. Here $C(r)$ represents the covariance between two model grid values, with r as the separation distance between the two grid points, and b is referred to as the correlation length scale. The parameter a represents the error variance of the model values.

The matrix multiplication, in (6), amounts to smoothing a field g with a Gaussian function when Σ_m based on the Gaussian function (7). Apart from the data assimilation algorithm, such smoothing is often performed in physical science applications.

3. LAPLACIAN SMOOTHING

The brief details given below are based on a method due to Derber (1990, a personal communication). Accordingly, The Gaussian smoothing of a gridded field could be achieved in a manner similar to when Σ_m and g were continuous. A Gaussian covariance function in a two dimensional Cartesian coordinates is given by

$$C(x, y) = a \exp[-(x^2 + y^2)/b^2] \quad (11)$$

A continuous function $g(x, y)$ can be expanded in the two-dimensional wave number Fourier spectral decomposition:

$$g(x, y) = \sum_{m=0}^{\infty} \sum_{n=0}^{\infty} a_{mn} \exp[-i(mx + ny)] \quad (12)$$

Then the effect of matrix multiplication is approximated by

$$\begin{aligned} g_C(0, 0) &= \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} g(x, y) C(x, y) dx dy \\ &= a\pi b^2 \sum_{m=0}^{\infty} \sum_{n=0}^{\infty} a_{mn} \exp \left[-\frac{(n^2 + m^2)b^2}{4} \right]. \end{aligned}$$

This is easily seen to be approximated by:

$$g_C(0, 0) \simeq \left(1 + \frac{\delta}{N} \nabla^2\right)^N g(0, 0) \quad (13)$$

where ∇^2 is the Laplacian operator, and N is large. Thus, the multiplication of Σ_m , which is based on a Gaussian covariance function, is approximated by an application of a large number of Laplacian operations at each model grid point.

As stated earlier, this algorithm is computationally quite fast as compared to the actual matrix multiplication, but it requires further enhancement for it to be practical. However, it suggests a new direction to view the problem in the spectral domain that leads to our enhancement technique.

4. POLYNOMIAL OPERATOR SMOOTHING

4.1 Derivation of Canonical Forms

Let $f(m, n)$ be the obtained by smoothing the gridded field $g(m, n)$ with a Gaussian weighting function. We will like to show that $f(m, n)$ can be approximated by successively applying polynomial operators, $P(D_1)$ and $P(D_2)$ where $D_j, j = 1, 2$ are simple averaging operators, given by

$$D_1 g(m, n) = [g(m+1, n) + g(m-1, n)]/2, \quad (14a)$$

$$D_2 g(m, n) = [g(m, n+1) + g(m, n-1)]/2. \quad (14b)$$

The implementation is affected by comparison of two canonical expressions which are quite similar in form. The first one is obtained by Fourier transform of a convolution, and the second is obtained by applying the polynomial product operator, $P(D_1)P(D_2)$ to the Fourier representation of $g(m, n)$.

Theorem 1. Let $\hat{g}(\theta, \phi)$ be the two dimensional Fourier transform of $g(m, n)$. Then

$$f(m, n) = \int_0^1 \int_0^1 \hat{g}(\theta, \phi) q(\theta) q(\phi) e^{2\pi i(m\theta + n\phi)} d\theta d\phi \quad (15)$$

where

$$q(\theta) = \sum_s \exp(-2\pi i s \theta) \exp(-\delta^2 s^2 / b^2), \quad (16)$$

and δ is the uniform distance between two consecutive x 's and y 's.

Theorem 2. Let $D_j, j = 1, 2$ be the averaging operators from (14). Then

$$\begin{aligned} P_2(D_2)P_1(D_1)g(m, n) &= \\ \int_0^1 \int_0^1 \hat{g}(\theta, \phi) P_2(\cos 2\pi\phi) P_1(\cos 2\pi\theta) e^{2\pi i(m\theta + n\phi)} d\theta d\phi. \end{aligned} \quad (17)$$

Note that the right hand sides of (16) and (17) are similar in functional form. Also, an examination of the two left hand sides indicates that we could approximate $f(m, n)$ as

$$f(m, n) \simeq P_2(D_2)P_1(D_1)g(m, n) \quad (18)$$

provided $P_1(\cos 2\pi\theta) \simeq q(\theta)$, and $P_2(\cos 2\pi\phi) \simeq q(\phi)$. In fact, since P_1 and P_2 approximate the same function q , it follows that $P_1 = P_2 = P \simeq q$. How good an approximation (18) is depends on how well we can approximate q by a polynomial over the domain of q .

In the following, we first prove the two results in (15) and (17).

Lemma 1. $f(m, n)$ is a convolution of $g(m, n)$ with a two-dimensional array $\sigma(k, l)$, given by

$$\sigma_m(k, l) = a \exp \left[-\frac{\delta^2 k^2 + \delta^2 l^2}{b^2} \right]. \quad (19)$$

Proof. Note that $f(m, n)$ is obtained from $\Sigma_m g$ as

$$f(m, n) = \sum_{s, t} g(s, t) a \exp \left[-\frac{(x_m - x_s)^2 + (y_n - y_t)^2}{b^2} \right] \quad (20)$$

Since $g(m, n)$ is defined on a uniform grid such that $x_m - x_s = \delta(m - s)$ and $y_n - y_t = \delta(n - t)$, (20) can be written as

$$\begin{aligned} f(m, n) &= \sum_{s,t} g(s, t) a \exp \left[-\frac{\delta^2(m-s)^2 + \delta^2(n-t)^2}{b^2} \right] \\ &= \sum_{s,t} g(s, t) \sigma(m-s, n-t) \\ &= g * \sigma, \end{aligned} \quad (21)$$

where $*$ indicates convolution.

The proof of Theorem 1 is an immediate consequence of the convolution (21).

Proof of Theorem 1. Denoting the Fourier transform by a hat, we get from (21),

$$\hat{f} = \widehat{g * \sigma} = \hat{g} \hat{\sigma}. \quad (22)$$

Also, from (19), we get

$$\hat{\sigma}(\theta, \phi) = q(\theta)q(\phi),$$

where $q(\theta)$ is given by (16). Theorem 1 is proved on taking the inverse Fourier transform of (22).

Proof of Theorem 2. With \hat{g} as the Fourier transform of g , we can write

$$g(m, n) = \int_0^1 \int_0^1 \hat{g}(\theta, \phi) e^{2\pi i(m\theta + n\phi)} d\theta d\phi. \quad (23)$$

Applying operator D_1 from (14) to (23), we get

$$\begin{aligned} D_1 g(m, n) &= \int_0^1 \int_0^1 \hat{g}(\theta, \phi) \frac{(e^{2\pi\theta} + e^{-2\pi\theta})}{2} e^{2\pi i(m\theta + n\phi)} d\theta d\phi \\ &= \int_0^1 \int_0^1 \hat{g}(\theta, \phi) \cos(2\pi\theta) e^{2\pi i(m\theta + n\phi)} d\theta d\phi. \end{aligned}$$

We note that, every time D_1 operates on g , we get an extra factor of $\cos(2\pi\theta)$ inside the integral. Thus, computing D_1, D_1^2, \dots , it is easy to see that for a polynomial operator $P_1(D_1)$, we obtain:

$$P_1(D_1)g(m, n) = \int_0^1 \int_0^1 \hat{g}(\theta, \phi) P_1(\cos 2\pi\theta) e^{2\pi i(m\theta + n\phi)} d\theta d\phi.$$

A similar application of operator polynomial $P_2(D_2)$ yields

$$P_2(D_2)g(m, n) = \int_0^1 \int_0^1 \hat{g}(\theta, \phi) P_2(\cos 2\pi\phi) e^{2\pi i(m\theta + n\phi)} d\theta d\phi.$$

Thus, a successive application of two polynomial operators, P_1 and P_2 , gives the desired expression (17), proving Theorem 2.

4.2 Polynomial Approximation

The above analysis indicates that an application of polynomial operators will approximate matrix multiplication provided we appropriately choose $P_1(\theta) \simeq q(\theta)$, and $P_2(\phi) \simeq q(\phi)$. First we note that $P_1 = P_2$, since P_1 and P_2 are to approximate the same function, $q(\cdot)$. Next consideration is determination of the polynomial P .

Following a method suggested in deBoor (1980), P is chosen to be the interpolating polynomial of degree n for the function q at the expanded Chebychev points $x_i, i = 1, \dots, n$, on the interval $[a, b]$ (deBoor, 1980)

$$x_i = \frac{1}{2} \left[a + b + (a - b) \frac{\cos \left(\frac{\pi(2i+1)}{2n+2} \right)}{\cos \left(\frac{\pi}{2n+2} \right)} \right]. \quad (22)$$

Here $[a, b]$ is the domain of the function $q(\theta)$. Rewriting (16) as

$$g(\theta) = 1 + 2 \sum_{s>0} \cos 2\pi s\theta e^{-s^2/b^2},$$

we note that $q(\theta)$ involves $\cos(2\pi s\theta)$ terms, for different integers, $s (> 0)$. Using the recursion relation

$$\cos(2\pi(s+1)\theta) = 2 \cos 2\pi\theta \cos 2\pi s\theta - \cos(2\pi(s-1)\theta).$$

we note that $q(\theta)$ is a function of $t = \cos 2\pi\theta$ with domain $[-1, 1]$.

5. NUMERICAL SIMULATIONS

The three Gaussian smoothing algorithms – Laplacian smoothing by Derber, the Polynomial Operator smoothing algorithm, and the actual matrix multiplication (6) were used in an actual data assimilation application. For the polynomial operator algorithm, we found that polynomials of degree around 8 are adequate to estimate the convolution product with error around 10^{-4} . This is for values of $\delta = 20$ and $b = 50$ kilometers. In general, the three algorithms provided results that were almost identical. However, our present method was about ten times computationally faster than that of Derber, and about 25 times faster than the actual matrix multiplication. We ran this on an actual data and on simulated data.

6. CONCLUDING REMARKS

Our estimate is computationally efficient and accurate. These results suggest a fairly general algorithm for estimating convolution products in general. The key is to give Fourier analysis and find a polynomial in the simple averaging operators that estimate the Fourier transform of the fixed element in the convolution product. If we can find a low degree polynomial that estimates this transform, then one will have an efficient and accurate method of computing the convolution product. In our application this gives an improvement in speed, and maintained the accuracy.

Our algorithm is quite general that it is not limited to Gaussian smoothing alone. It is applicable to other data assimilation applications where the covariance matrix Σ_m in (6) is based on other correlation structures; only restriction is that the correlation function be a product of even functions of one variable. An immediate extension of this then is the smoothing function

$$w(r) = w(x, y) = \exp \left[-\frac{x^2}{b_x^2} - \frac{y^2}{b_y^2} \right] \quad (23)$$

which is a product of two Gaussian functions, and has different correlation length scales b_x and b_y in X and Y directions. In fact, smoothing function (23) tacitly allows for non-uniform (different) grid lengths in the two directions. Obviously, this is easily generalized to more than two dimensions. Such applications along with details of rigor will be described elsewhere.

ACKNOWLEDGMENTS

We would like to thank Ms. Hope Hamilton of the National Center for Atmospheric Research for providing us with the T_EX macros for the two-column format used in this paper.

REFERENCES

- deBoor, C., 1980: *Elementary Numerical Analysis*. McGraw Hill, New York.
- Derber, J., and A. Rosati, 1989: A global data assimilation system. *J. Phys. Ocean.*, 1333-1347.
- Navon, I.M., and D.M. Legler, 1987: Conjugate-gradient methods for large-scale minimization in meteorology. *Mon. Wea. Rev.*, 115, 1479-1502.
- Passi, R.M., R.K. Goodrich, M. Limber, 1992: A convolution algorithm with application to data assimilation. To be submitted to *SIAM J. Appl. Math.*
- Passi, R.M., R.K. Goodrich, M. Limber, and J. Derber, 1992: Performance evaluation of three data assimilation algorithms with Gaussian covariance structure for the model output. To be published as an *Institute for Naval Oceanography Technote*.

Computing Quantile Smoothing Splines

Roger W. Koenker
Department of Economics
University of Illinois at Urbana-Champaign
Champaign, IL 61820

Pin T. Ng
Department of Economics
University of Houston
Houston, TX 77204-5882

Abstract

We adapt and modify the algorithm of Barrodale and Roberts (1973) to compute the quantile smoothing splines defined as solutions to

$$\min_{g \in C^1} \sum \rho_\alpha(y_i - g(x_i)) + \lambda \int_0^1 |g''(x)| dx$$

with $\rho_\alpha(u) = (\alpha - I(u < 0))u$ and $0 \leq \alpha \leq 1$. Quantile smoothing splines provide a general approach to nonparametric estimation of conditional quantile functions, just as classical smoothing splines offer a general approach to estimating conditional mean functions. The entire path of quantile smoothing splines for a given penalty parameter λ as well as all the distinct solutions corresponding to distinct λ for a given quantile α can be found by performing sequences of simplex pivots following the approach in parametric linear programming.

1. Introduction

In practice nonparametric regression is virtually synonymous with the estimation of flexible models for conditional mean functions. Koenker and Ng [4] suggested, as an alternative measure of conditional central tendency, median smoothing splines which estimate the conditional median function of y given x . Aspects other than the central tendency of conditional distributions are, however, frequently of substantial interest. For examples, we might consider estimating the seasonal pattern of extreme temperatures, water levels, or pollution readings. Are estimated trends in mean incomes or SAT performance consistent with trends in extreme quantiles? Quantile smoothing splines provide one way to answer this question.

Koenker and Ng [4] showed that the problem of

$$\min_{g \in C^1[0,1]} \sum_{i=1}^n \rho_\alpha(y_i - g(x_i)) + \lambda \int_0^1 |g''(x)| dx \quad (1.1)$$

where $C^1[0,1]$ is the space of continuous functions on $[0,1]$ with continuous first derivative, $\rho_\alpha(u) = (\alpha - I(u < 0))u$ and $0 \leq \alpha \leq 1$ retains the linear programming

form of the linear parametric version of the quantile regression problem, and the solution, i.e. the α th quantile smoothing spline, is a parabolic spline. The α th quantile smoothing spline divides the observations on y into two parts with roughly αn of them falling below or on the fitted spline and $(1 - \alpha)n$ of them above. These quantile smoothing splines, unlike the classical l_2 smoothing spline, are qualitatively robust to gross errors in the observations on y . As the smoothing parameter $\lambda \rightarrow 0$, the solution becomes the quantile interpolating spline while for λ sufficiently large it yields the bivariate linear regression quantile estimate. Following the approach of parametric linear programming, the whole family of quantile smoothing splines for a fixed λ as well as the entire path of solutions in λ for a given quantile α can be easily found.

In this paper, we detail the computation of the quantile smoothing splines and the parametric linear programming in finding the whole sequences of λ and α through adapting and modifying the well known simplex algorithm of Barrodale and Roberts [1] for l_1 regression.

2. Theory

Given observations $\{(y_i, x_i) : i = 1, \dots, n\}$ with $0 < x_1 < \dots < x_n < 1$, we define the α th quantile smoothing spline, $g_{\alpha,\lambda}$, as the solution to (1.1). Koenker and Ng [4] showed that there exists a quadratic spline of the form

$$\hat{g}_{\alpha,\lambda}(x) = \alpha_i(x - x_i)^2 + \beta_i(x - x_i) + \gamma_i \quad \text{for } x_i \leq x < x_{i+1}, \quad i = 0, \dots, n$$

which solves (1.1). Schuette [5] studies a closely related problem with a discrete penalty for equally spaced observations.

Let $h_i = x_{i+1} - x_i$, for $i = 1, \dots, n-1$. From the continuity constraints of the quadratic spline, we have

$$\hat{g}(x_{i+1}) = \alpha_i h_i^2 + \beta_i h_i + \gamma_i = \gamma_{i+1} \quad (2.1)$$

$$\hat{g}'(x_{i+1}) = 2\alpha_i h_i + \beta_i = \beta_{i+1}$$

$\beta_0 = \beta_1$, $\gamma_0 = \gamma_1$, for $i = 1, \dots, n-1$. Note that the quadratic spline must be linear in the exterior intervals $[0, x_1]$ and $(x_n, 1]$ otherwise the "roughness" term

in (1.1) could be reduced without affecting the "fidelity" portion. This gives us $\alpha_0 = \alpha_n = 0$. Eliminating the β_i 's in (2.1) yields

$$\alpha_i h_i + \alpha_{i+1} h_{i+1} = \left[\frac{\gamma_{i+2} - \gamma_{i+1}}{h_{i+1}} \right] - \left[\frac{\gamma_{i+1} - \gamma_i}{h_i} \right]$$

$$i = 1, \dots, n-2$$

We have $3(n+1)$ free parameters and $2(n+1)$ constraints, which leaves us with $n+1$ free parameters.

Letting $\theta' = (\alpha_1, \gamma_1, \dots, \gamma_n)$ be the $n+1$ free parameters and noticing that the roughness term in (1.1) can be written as

$$\int_0^1 |g''(x)| dx = 2 \sum_{i=1}^{n-1} h_i |\alpha_i|,$$

we may rewrite (1.1) as

$$\min_{\theta \in R^{n+1}} \sum_{i=1}^n \left[\frac{1}{2} + \left(\alpha - \frac{1}{2} \right) \operatorname{sgn}(\tilde{y}_i - \tilde{x}_i \theta) \right] |\tilde{y}_i - \tilde{x}_i \theta|$$

$$+ \sum_{i=n+1}^{2n-1} |\tilde{y}_i - \lambda \tilde{x}_i \theta| \quad (2.2)$$

where $\tilde{y}' = (y', 0'_{n-1})$,

$$\tilde{X} = \begin{bmatrix} 0 & & & \\ \vdots & I_n & & \\ 0 & & & \\ \dots & & & \\ A & & & \end{bmatrix}_{(2n-1) \times (n+1)}$$

$A = HD^{-1}B$, $H = \operatorname{diag}(h)$, D a $(n-1) \times (n-1)$ banded matrix with

$$d_{1,1} = 1, \quad d_{j,j} = h_j, \quad d_{j,j-1} = h_{j-1}$$

for $j = 2, \dots, n-1$, and B a $(n-1) \times (n+1)$ banded matrix with

$$b_{1,1} = 1, \quad b_{j,j} = h_j^{-1}, \quad b_{j,j+1} = -(h_j^{-1} + h_{j+1}^{-1})$$

$$b_{n-1,n+1} = h_{n-1}^{-1}, \quad j = 2, \dots, n-1.$$

Notice that the objective function (2.2) can be expressed as a linear programming problem

$$\min_{u,v,b,c} R_{\alpha,\lambda}^*(u,v,b,c) =$$

$$\sum_{i=1}^n \left[1 + (2\alpha - 1) \operatorname{sgn} \left(\tilde{y}_i - \sum_{j=1}^{n+1} (b_j - c_j) \tilde{x}_{ij} \right) \right]$$

$$(u_i + v_i) + \lambda \sum_{i=n+1}^{2n-1} (u_i + v_i) \quad (2.3)$$

$$\text{s.t. } \tilde{y}_i = \lambda_i \sum_{j=1}^n (b_j - c_j) \tilde{x}_{ij} + u_i - v_i$$

$$\text{and } b_j, c_j, u_i, v_i \geq 0$$

$$\text{for } i = 1, \dots, 2n-1, \text{ and } j = 1, \dots, n$$

in which $\lambda_i = I(i \leq n) + \lambda I(i > n)$. With this linear programming formulation, we can then modify the efficient simplex algorithm of Barrodale and Roberts [1] to solve for u, v, b, c and hence θ .

Following the parametric linear programming approach, the whole family of quantile smoothing splines for $\lambda \in [0, \infty)$, and the whole path of solutions in the smoothing parameter λ for a given α may be efficiently computed.

3. Implementation

3.1. The Simplex Method

Denote $\epsilon_i = u_i - v_i$. The marginal costs for $\{b, c, u, v\}$ in (2.3) are

$$\frac{\partial R_{\alpha,\lambda}^*}{\partial b_j} = \sum_{i=1}^n [1 + (2\alpha - 1) \operatorname{sgn}^*(\epsilon_i; -\tilde{x}_{ij})]$$

$$\operatorname{sgn}^*(\epsilon_i; -\tilde{x}_{ij})(-\tilde{x}_{ij})$$

$$+ \lambda \sum_{i=n+1}^{2n-1} \operatorname{sgn}^*(\epsilon_i; -\tilde{x}_{ij})(-\tilde{x}_{ij})$$

$$\frac{\partial R_{\alpha,\lambda}^*}{\partial c_j} = -\frac{\partial R_{\alpha,\lambda}^*}{\partial b_j}$$

$$\frac{\partial R_{\alpha,\lambda}^*}{\partial u_i} = 2\alpha I(\epsilon_i \geq 0) I(i \leq n)$$

$$+ \lambda I(\epsilon_i \geq 0) I(i > n)$$

$$\frac{\partial R_{\alpha,\lambda}^*}{\partial v_i} = (2 - 2\alpha) I(\epsilon_i < 0) I(i \leq n)$$

$$+ \lambda I(\epsilon_i < 0) I(i > n)$$

where

$$\operatorname{sgn}^*(w; z) = \begin{cases} \operatorname{sgn}(w) & \text{if } w \neq 0 \\ \operatorname{sgn}(z) & \text{if } w = 0 \end{cases}$$

With the objective function expressed as a linear programming problem along with the marginal costs of the coefficients, we can then adopt and modify the algorithm of Barrodale and Roberts [1]. We only describe the following modifications to make the algorithm work in our setting. Readers are referred to [1] for the details and theory of the original linear programming algorithm.

1. The initial full simplex tableau for (2.3) is as follows, in which

$$m_{1,j} = m_{2,j} + m_{3,j}\lambda + m_{4,j}\alpha$$

$$m_{2,j} = \sum_{i=1}^n [1 - \operatorname{sgn}^*(\epsilon_i; \tilde{x}_{ij})]$$

$$\operatorname{sgn}^*(\epsilon_i; -\tilde{x}_{ij}) \tilde{x}_{ij}$$

Basis	R	b_1	...	b_{n+1}	c_1	...	c_n
u_1	\tilde{y}_1	$x_{1,1}$...	$x_{1,n}$	$-x_{1,1}$...	$x_{1,n}$
\vdots	\vdots	\vdots		\vdots	\vdots		\vdots
u_n	\tilde{y}_n	$x_{n,1}$...	$x_{n,n+1}$	$-x_{n,1}$...	$-x_{n,n+1}$
u_{n+1}	\tilde{y}_{n+1}	$x_{n+1,1}$...	$x_{n+1,n+1}$	$-x_{n+1,1}$...	$-x_{n+1,n+1}$
\vdots	\vdots	\vdots		\vdots	\vdots		\vdots
u_{2n-1}	\tilde{y}_{2n-1}	$x_{2n-1,1}$...	$x_{2n-1,n+1}$	$-x_{2n-1,1}$...	$-x_{2n-1,n+1}$
m_1		$m_{1,1}$...	$m_{1,n+1}$	$-m_{1,1}$...	$-m_{1,n+1}$
m_2		$m_{2,1}$...	$m_{2,n+1}$	$-m_{2,1}$...	$-m_{2,n+1}$
m_3		$m_{3,1}$...	$m_{3,n+1}$	$-m_{3,1}$...	$-m_{3,n+1}$
m_4		$m_{4,1}$...	$m_{4,n+1}$	$-m_{4,1}$...	$-m_{4,n+1}$

Basis	R	u_1	...	u_n	u_{n+1}	...	u_{2n-1}	v_1	...	v_n	v_{n+1}	...	v_{2n-1}
u_1	\tilde{y}_1	1						-1					
\vdots	\vdots		\ddots						\ddots				
u_n	\tilde{y}_n			1						-1			
u_{n+1}	\tilde{y}_{n+1}				1						-1		
\vdots	\vdots					\ddots						\ddots	
u_{2n-1}	\tilde{y}_{2n-1}						1						-1
m_1		-2α	...	-2α	$-\lambda$...	$-\lambda$	$(-2+2\alpha)$...	$(-2+2\alpha)$	$-\lambda$...	$-\lambda$
m_2		0	...	0	0	...	0	-2	...	-2	0	...	0
m_3		0	...	0	-1	...	-1	0	...	0	-1	...	-1
m_4		-2	...	-2	0	...	0	2	...	2	0	...	0

$$m_{3,j} = \sum_{i=n+1}^{2n-1} \text{sgn}^*(\epsilon_i; -\tilde{x}_{ij}) \tilde{x}_{ij}$$

$$m_{4,j} = 2 \sum_{i=1}^n \text{sgn}^*(\epsilon_i; \tilde{x}_{ij}) \tilde{x}_{ij}$$

where $m_{3,j}$ and $m_{4,j}$ are the coefficients of λ and α in $m_{1,j}$ respectively while $m_{2,j}$ is the term independent of both. The negative marginal cost m_1 is broken down into three pieces mainly to facilitate the computation of "next λ " and "next α " described in the next section.

2. The condensed tableau is stored in a $(2n+4) \times (n+3)$ array. The first $(2n-1)$ rows and $(n+1)$ columns store the \tilde{x}_{ij} . The first $(2n-1)$ rows of the $(n+2)$ th column store the \tilde{y}_i . The first $(2n-1)$ rows of the $(n+3)$ th column store the labels of the initial basis u_i as $(n+1)+i$. If v_i enters as the basis, the label becomes $-(n+1)-i$. The labels for the initial non-basis b_j are stored in the $((2n-1)+5)$ th row as j . The $((2n-1)+2)$ th, $((2n-1)+3)$ th, $((2n-1)+4)$ th and $((2n-1)+1)$ th rows store the $m_{2,j}$, $m_{3,j}$, $m_{4,j}$ and $m_{1,j}$ respectively.
3. During Stage 1 the vector chosen to enter the basis is the one among b_j and c_j that has the smallest

negative marginal cost and during Stage 2 is the one among the nonbasic vectors u_i and v_i with the smallest negative marginal cost as in [1]. Once a b_j or c_j is entered into the basis, it is not allowed to leave the basis. However, notice here that the sum of the negative marginal costs of each pair of b_j and c_j is zero while the sum of the negative marginal costs of each pair of u_i and v_i is -2 if $i \in H_1$ and -2λ if $i \in H_2$, where $H_1 = \{1, \dots, n\}$ and $H_2 = \{n+1, \dots, 2n-1\}$ corresponding to the indices in the "fidelity" and "roughness" portions of (2.3) respectively.

4. As in [1], the vector chosen to leave the basis in both Stage 1 and Stage 2 is the one among the basic vectors u_i and v_i which causes the maximum reduction in the objective function. The modified simplex transformation rule in [1] is further modified as follows. First apply the normal rule to determine the pivot. If subtracting twice the value of the pivot from $m_{1,j}$ yields a nonpositive number, proceed to the simplex transformation. Otherwise, if the pivot belongs to H_1 , subtract twice the pivotal row from the $m_{1,j}$ row and the $m_{2,j}$ row, multiply the pivotal row by (-1) , and replace in the basis the vector u_i or (v_i) corresponding to the pivotal row by v_i or (u_i) . If the pivot belongs to H_2 , subtract twice the

pivotal row multiplied by λ from the $m_{1,j}$ row and twice the pivotal row from the $m_{3,j}$ row, multiply the pivotal row by (-1) , and replace the basic vector u_i or (v_i) corresponding to the pivotal row by v_i or (u_i) . This operation is repeated until eventually a chosen pivot cannot be rejected and the usual simplex transformation is then carried out. This movement through several neighboring simplex vertices in a single pass has been shown to be much more efficient than the earlier versions of simplex transformation for l_1 type problems.

3.2. Parametric Linear Programming

Since the simplex algorithm terminates when there is no nonbasic u_i or v_i with negative marginal cost and with the sum of the negative marginal costs for each pair of u_i and v_i being -2 if $i \in H_1$ and being -2λ if $i \in H_2$, we can easily perform the following parametric linear programming on λ and α .

3.2.1 Next λ

To compute the entire path of solutions in λ for a given α , we start from $\lambda_0 = 0$, the α th quantile interpolating spline. The solution $\hat{g}_{\alpha, \lambda_0}$ remains optimal if and only if

$$-2 \leq m_{2,j} + m_{3,j}\lambda + m_{4,j}\alpha \leq 0, \quad j \in H_1 \quad (3.1)$$

$$-2\lambda \leq m_{2,j} + m_{3,j}\lambda + m_{4,j}\alpha \leq 0, \quad j \in H_2$$

Hence, the next value of λ at which $\hat{g}_{\alpha, \lambda}$ cease to be optimal is

$$\lambda_1 = \min_{\lambda > \lambda_0} \left\{ \min_{j \in H_1} \left[-\frac{2 + m_{2,j} + m_{4,j}\alpha}{m_{3,j}} \right], \min_{j \in H_2} \left[-\frac{m_{2,j} + m_{4,j}\alpha}{2 + m_{3,j}} \right], \min_{j \in H_2} \left[-\frac{m_{2,j} + m_{4,j}\alpha}{m_{3,j}} \right] \right\}$$

Continuing this iteration until $\lambda_{m+1} = \lambda_m$ will yield all the distinct α th quantile smoothing splines with the solution for $\lambda \geq \lambda_m$ corresponding to the linear α th regression quantile fit to the observations.

3.2.2 Next α

Similarly the entire family of quantile smoothing splines for a fixed λ can be obtained by starting the iteration at $\alpha_0 = 1/n$, which yields the 0th quantile smoothing spline, as in Koenker and d'Orey [3]. The solution $\hat{g}_{\alpha_0, \lambda}$ remains optimal if and only if the constraints on the marginal costs as stated in 3.1 are satisfied. Hence, the next α is

$$\alpha_1 = \min_{\alpha > \alpha_0} \left\{ \min_{j \in H_1} \left[-\frac{2 + m_{2,j} + m_{3,j}\lambda}{m_{4,j}} \right], \min_{j \in H_2} \left[-\frac{2 + \frac{m_{2,j}}{\lambda} + m_{3,j}}{\frac{m_{4,j}}{\lambda}} \right], \min_{j \in H_2} \left[-\frac{\frac{m_{2,j}}{\lambda} + m_{3,j}}{\frac{m_{4,j}}{\lambda}} \right] \right\}$$

$$\left. -\frac{m_{2,j} + m_{3,j}\lambda}{m_{4,j}} \right\}, \min_{j \in H_2} \left[-\frac{2 + \frac{m_{2,j}}{\lambda} + m_{3,j}}{\frac{m_{4,j}}{\lambda}} \right], \min_{j \in H_2} \left[-\frac{\frac{m_{2,j}}{\lambda} + m_{3,j}}{\frac{m_{4,j}}{\lambda}} \right] \right\}$$

Again the iteration is carried out until $\alpha_{p+1} = \alpha_p$ which gives the 1.0th quantile smoothing spline.

4. An Example

To illustrate the methods discussed above Figure 4.1 depicts three quantile smoothing splines for the well-known motorcycle data, see Härdle [2]. Obviously, the estimated quantiles reflect a substantial degree of non-homogeneity in the shape of the conditional distribution of the response at the various time coordinates.

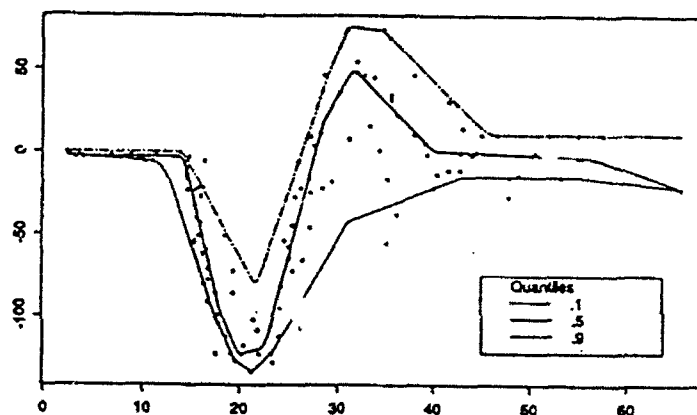


Figure 4.1: Three Quantile Smoothing Splines for the Motorcycle Data

References

- [1] Barrodale, I. and F.D.K. Roberts, (1973), "An Improved Algorithm for Discrete l_1 Linear Approximation", *SIAM J. Numer. Anal.*, 10, 839-848.
- [2] Härdle, W. (1989), *Applied Nonparametric Regression*, New York: Cambridge University Press.
- [3] Koenker, R. and V. d'Orey, (1987), "Computing Regression Quantiles", *Applied Statistics*, 383-393.
- [4] Koenker, R. and P. Ng, (1992), "Quantile Smoothing Splines", in *Nonparametric Statistics and Related Topics* (ed: A.K.Md.E. Saleh), (North-Holland: New York).
- [5] Schuette, D. R. (1978), "A linear programming approach to graduation", *Trans. Soc. of Actuaries*, 30, 407-445.

A New Nonparametric Estimation Method: Local and Nonlinear

Andrzej S. Kozek*

Institute of Computer Science, University of Wrocław, Poland
Przesmycki Str. 20, 51-151 Wrocław, Poland

Abstract

In the paper we consider a new class (called NoLoess) of nonparametric estimators of regression function $r(x) = E(Y | X = x)$. The new estimators link nonparametric and parametric methods. NoLoess extends the class of Macauley, Cleveland, and Stone estimators and replaces their locally linear models (Loess) given by (1.3) with a general locally parametric nonlinear model $g(x, \theta)$. The parameter θ is estimated locally by fitting $g(x, \theta)$ to a sub-sample $(X_{i_1}, Y_{i_1}), \dots, (X_{i_{k(x)}}, Y_{i_{k(x)}})$ with $X_{i_1}, \dots, X_{i_{k(x)}}$ being close to x . The resulting nonparametric estimator of $r(x)$ is of the form $g(x, \hat{\theta}(x))$. Under proper regularity assumptions it is strongly consistent.

An interactive software package FS 2.0 running under MS DOS on IBM PC and compatible, and implementing a variety of nonparametric Loess- and NoLoess-type estimators has been developed.

1. Introduction

The importance of the use of the correct model or its satisfactory approximation is well recognized in Statistics. In this direction, in 1990 Parzen proposed a unification of methods in statistical data analysis. His four step scheme for the model identification process is based both on parametric and nonparametric methods, and on their comparison. Consistent tests of fit of parametric models for regression function $r(x) = E(Y | X = x)$ have also been proposed in Kozek (1990, 1991). The tests are based on a comparison of parametric and nonparametric estimators of $r(x)$. For tests and discussion on similar subject in case of non-random design x we refer to Eubank and Spiegelman (1990).

In the present note we follow the idea of comparison of parametric and nonparametric estimators and propose an extension of nonparametric methods for estimation of a regression function. Here we consider only the random design. Let (X, Y) be a $(p + 1)$ -dimensional random vector (r.v.), $X \in R^p$, $Y \in R^1$ and $(X, Y), (X_1, Y_1), \dots, (X_n, Y_n)$ be independent identically distributed (i.i.d.) r.v. It is well known (cf.

Härdle et al. (1988) and Hall and Jones (1990)) that the most frequently considered nonparametric estimators such as Nadaraya-Watson (NW), k-Nearest Neighbor (NN), and p-Optimal Quantile (OQ) (see Kozek and Schuster (1990) for more details) are minimizing a local fit function

$$\mathcal{M}_0(\theta) = \sum_{i=1}^n (Y_i - \theta)^2 K\left(\frac{x - X_i}{h_v}\right), \quad (1.1)$$

where suitable window bandwidth h_v is specific for a given estimator type, and where $K(\cdot)$ is a nonnegative kernel.

We propose to extend (1.1) to the form

$$\mathcal{M}(\theta) = \sum_{i=1}^n M(Y_i - g(X_i, \theta)) K\left(\frac{x - X_i}{h_v}\right) W_M(X_i), \quad (1.2)$$

where

$g(x, \theta)$ is a 'trial' parametric regression function (in general g may be a nonlinear function of the vector parameter θ),

M is a convex function defining a conditional M-functional,

W_M is a nonnegative weight function.

The estimators of conditional M-functionals are of the form $g(x, \hat{\theta}(x))$, where $\hat{\theta}(x)$ is the value of θ which minimizes (1.2). They extend several important classes of nonparametric estimators considered so far. We mentioned already NW, NN, and OQ estimators of the regression function which are of the form (1.2) with $M(y) = y^2$ and $g(x, \theta) = \theta$, $\theta \in R^1$. If $M(y) = y^2$ and

$$g(x, \theta) = \alpha_0 + \alpha \cdot x \quad (1.3)$$

with \cdot standing for the usual inner product on R^p and $\theta = (\alpha_0, \alpha_1, \dots, \alpha_p) = (\alpha_0, \alpha)$ then we get the class of estimators locally linear in parameters, called Loess (Cleveland (1979)) and introduced by Macauley (1931), Stone (1977), and Cleveland (1979). Since our estimators can be considered as nonlinear Loess we shall use abbreviation NoLoess in the sequel.

Case of $\theta \in R^1$ and a convex function M with a bounded derivative admits an easy interpretation. Then $M'(y) = c \cdot (G(y) - \pi)$, where G is a cumulative distribution function (c.d.f.) of some r.v. Z , and the integral

*The author is visiting thru academic year 1991-1992 at The University of Arkansas, Fayetteville.

$\int M(y - \theta)F(dy)$ achieves its minimum at a $(1 - \pi)$ -quantile of the c.d.f. of the r.v. $Y - Z$, where $Y \sim F$ (Kozek (1984), p. 155). If the probability distribution of Y has a symmetry center and M is symmetric about zero then the minimum of the integral coincides with the symmetry center, median, and expectation of F . Clearly, this coincidence fails in general when F is not symmetric. Since M is used to make the corresponding estimator robust and no deviation from expectation is acceptable, the conditional M-functionals require additional symmetry assumption on the conditional probability distribution of Y given $X = x$. Estimators of functionals of conditional probability distributions have been already considered in the literature, cf. Härdle et al. (1988), Hall and Jones (1990), and the references quoted in these papers.

Our experience with software packages FS and FS 2.0¹ shows that Loess type estimators are typically performing much better in small and moderate sample sizes than NW, NN, and OQ estimators do. This should not be surprising since in case of a box-type kernel NW, NN, and OQ are approximating $r(x)$ using piecewise constant functions while Loess uses locally a linear space of functions, e.g. polynomials. Smooth kernels are smoothing the resulting fit and 'make up' the lack of any formal smoothness requirement present in case of splines. Formal justification of better performance of Loess estimators in terms of the reduction of the Mean Squared Error (MSE) can be found e.g. in Fan (1991). NW, NN, OQ and Loess lack however the superb feature of parametric models: the ease of parameter interpretation. We hope that NoLoess and interpretation of the dependence of parameters $\theta(x)$ on x may help researchers to improve nonadequate parametric models or, in 'weak or no dependence' case, may support the model and its interpretation. We recommend for $g(x, \theta)$ a function which represents experts knowledge about the actual regression function. In this case only almost a parametric fit and typically small or moderate corrections through dependence of parameters on x are required.

2. Fit Short 2.0 — an Implementation of NoLoess

Fit Short 2.0 (FS 2.0) is an interactive package written in Borland's Turbo Pascal^{®2}. FS 2.0 implements NoLoess in the form given by (1.2) for $x \in R^1$ and with a broad range of options.

Smoothing parameter v_0 is called 'optimal' if it equals

¹FS and FS 2.0 were written by the author with an assistance of K. K. Kozek and use a system of windows written by J. Witkowski.

²Turbo Pascal is a registered trademark of Borland Int., Inc.

argmin of

$$\psi(v) = \sum_{i=1}^n \Psi(Y_i - g(X_i, \hat{\theta}_v)) \Xi * W_\Psi, \quad (2.1)$$

where Ψ meets the same conditions as M in (1.2),

$$\Xi * W_\Psi = \Xi \left(\frac{K(0)}{\sum K\left(\frac{x-X_i}{h}\right)} \right) \cdot W_\Psi(X_i),$$

and W_Ψ is a bounded and positive weight function.

User can either supply his own value of a smoothing parameter v or choose an 'automatic' selection of an optimal smoothing parameter v_0 . FS 2.0 provides a collection of penalizing functions Ξ including those listed in Härdle (1991), p.157 and also a *Fit Short Curve* option of optimality in the form considered in Kozek and Schuster (1991).

All calculations reported in Section 5 were obtained using FS 2.0. We postpone a more detailed description of the modular structure of the package and implemented numerical solutions to another paper.

3. Technical Requirements for NoLoess

In this section we list assumptions necessary for the consistency result from Section 4.

A. (X, Y) is a random vector in $\mathcal{X} \times R^1$, $\mathcal{X} \subset R^p$ such that

$$Y = r(X) + \varepsilon, \quad (3.1)$$

where r is bounded and has bounded and continuous derivatives on \mathcal{X} .

B. X and ε are independent, and ε has symmetric probability distribution; both X and ε are assumed to have probability density functions, f_X and f_ε respectively, which are differentiable and have bounded derivatives.

C. M is a symmetric strictly convex function on R^1 satisfying condition $M(2t) \leq C \cdot M(t)$ for all t , and with a continuous, bounded derivative M' .

D. $g(x, \theta)$ is bounded for every θ and has a continuous vector of derivatives $g_\theta(x, \theta)$, the euclidean norm of which has a bounded envelope on $\mathcal{X} \times \Theta$, $\Theta \subset R^q$.

E. For every $x \in \mathcal{X}$

$$(r(x) - \epsilon, r(x) + \epsilon) \subset \{\alpha \in R^1 : \alpha = g(x, \theta), \theta \in \Theta\}.$$

F. $K(x) = k(|x|)$, where k is differentiable, bounded, positive and decreasing on $[0, 1)$, and $k(1) = 0$. W_M and W_Ψ are bounded and positive.

G. $EM(\varepsilon) < \infty$, and $E(M'(\varepsilon))^2 < \infty$.

H. For every $x \in \mathcal{X}$ and $h > 0$ the function

$$\Lambda(x, \theta, h) =$$

$EM'(Y - g(X, \theta)) g_\theta(x, \theta) K((x - X)/h) W_M(X)$ has a unique zero at $\theta_0(x, h)$.

I. For every $0 < \epsilon < \epsilon_0$, and $0 < h < h_0$

$$\inf |\Lambda(x, \theta, h)| > c \cdot h^p \epsilon, \quad (3.2)$$

where the infimum is over $(x, \theta) \in \mathcal{X} \times B(\theta_0(x, h), \epsilon)$, and $B(\theta, \epsilon)$ stands for the ball in R^p with center at θ and radius ϵ .

4. Consistency of NoLoess

Theorem. Let $(X, Y), (X_i, Y_i), i = 1, \dots, n$ be independent random variables, and assume that the conditions stated in Section 3 are fulfilled. If

$$h_n \rightarrow 0, \text{ and } nh_n^d / (\log n) \rightarrow \infty,$$

and estimator $g(x, \hat{\theta}(x))$ satisfies

$$E_{P_n} M'(Y - g(X, \hat{\theta})) \cdot g_\theta(x, \hat{\theta}) K((x - X)/h_n) = 0, \quad (4.1)$$

where P_n is the empirical probability measure generated by $(X_i, Y_i), i = 1, \dots, n$ then

$$g(x, \hat{\theta}(x)) \rightarrow r(x) \text{ a.s.}$$

If $r(x) = g(x, \theta_0)$ for some θ_0 then the convergence holds also for a constant window bandwidth $h_n = h$ and for the constant kernel $K(x) \equiv 1$.

Proof. Let $\mathcal{F}^1 = \{M'(y - g(x, \theta)) \cdot g_\theta(x, \theta) : \theta \in \Theta\}$ and $\mathcal{F}^2 = \{K((z - x)/h_n) : z \in X\}$ be families of functions. It is easy to see that the conditions of Section 3 imply \mathcal{F}^1 and \mathcal{F}^2 have polynomial discrimination. Thus—by the triangle inequality—the family $\mathcal{F}_n = \{f_1 \cdot f_2 : f_1 \in \mathcal{F}^1, f_2 \in \mathcal{F}^2\}$ also has polynomial discrimination. By Approximation Lemma (Pollard (1984), p.27) we get a bound for the covering number

$$N_1(\epsilon, P, \mathcal{F}_n) \leq A\epsilon^{-W},$$

where constants A and W do not depend on $\epsilon \in (0, 1)$. Since for every function $f \in \mathcal{F}_n$ we have $|f| \leq \text{const} \cdot K(\frac{z-x}{h})$ we get $(E|f|^2)^{1/2} \leq \text{const} \cdot h_n^p$ (see Pollard (1984), pp. 35-36 for more details). Now, we infer from Theorem 37, Chapter 2 in Pollard (1984) that

$$\sup_{f \in \mathcal{F}_n} |E_{P_n} f - E_P f| \leq \text{const} \cdot h_n^p.$$

Let ψ stand for a function from \mathcal{F}^1 . By Lemma 1 in Greblicki et al. (1984) we get

$$\left(1/(nh_n^d)\right) \sum_{i=1}^n \psi(X_i, Y_i, \theta) K((x - X_i)/h) \xrightarrow{\text{a.s.}} E(\psi(X, Y, \theta) | X = x) \cdot f_X(x).$$

Suppose that estimator $g(x, \hat{\theta}(x))$ is not consistent with probability 1. Then assumption (3.2) on $\Lambda(x, \theta, h)$

and the convergence obtained above contradict property (4.1) of the estimator.

The consistency in case of the parametric model follows from Theorem 2 in Huber (1967) and the observation that under assumptions of Section 3 the function $EM(Y - g(X, \theta)) \cdot K(\frac{x-X}{h})$ attains for a.e. x its minimum at the same point θ_0 which is unique for every nonsingular probability distribution of X . Indeed, (3.1), independence of X and ϵ , and the Anderson Lemma imply that $g(x, \theta_0) = g(x, \hat{\theta}(x))$ for a.e. x , where $\hat{\theta}(x)$ minimizes the conditional expectation $E(M(Y - g(X, \theta)) \cdot K(\frac{x-X}{h}) | X = x)$. Now, (3.2) implies that $\theta_0 = \hat{\theta}(x)$ a.s. \diamond

5. Example

We applied NoLoess and FIT SHORT 2.0 for data from volatile organic compounds (VOC) emission considered in Dunn³ and Chao (1992). Dunn and Chao considered models based on systems of differential equations with linear terms which relate several factors. These authors obtained the corresponding analytic solutions and fitted the parameters by the nonlinear least squares method. The original fit was not satisfactory, but it was considerably improved with a proper system of weights.

In the remaining part of the paper we briefly report on our attempt to answer the question if simplified models with variable coefficients can be useful in modelling the VOC emission effects. One of the considered simplified models was given by a differential equation (d.e.)

$$(d/dt)c(x) = a - b \cdot c(x), \quad c(x_0) = d,$$

with solution for $x_0 = 48.0$ of the form

$$c(x) = A + B \exp(-C(x - 48.0)). \quad (5.1)$$

An alternative trial model was suggested by a linear fit for a log—log plot of data and was given by a d.e.

$$(d/dt)c(x) = a \cdot c^b(x), \quad b > 1.0, \quad c(x_0) = d.$$

Its solution is of the form

$$c(x) = A/(x + B)^C. \quad (5.2)$$

We let $\theta = (A, B, C)$, and use $g(x, A, B, C) = A + B \exp(-C(x - 48.0))$ in the 'exponential' case (5.1), and $g(x, A, B, C) = A/(x + B)^C$ in the 'power' case (5.2). Since the data are highly non-uniform we have chosen

³The author would like to thank Professor J.E. Dunn for calling his attention to VOC emission modelling problems and for permission to use in his research data from U.S. Environmental Protection Agency, Air and Energy Engineering Research Laboratory, Research Triangle Park, NC.

the OQ type of the window bandwidth.

Figure 1 shows behavior of parametric estimators ($K = \Psi = \Xi = W_{\Psi} = W_{\Xi} \equiv 1.0$) and nonparametric ones (OQ, $K(x) = c \cdot (1 - x^2)^2$, $M(x) = \Psi(x) = |x| - \ln(1 + |x|)$, $\Xi(u) = 1/(1 - u)^2$, $W_{\Psi} = W_{\Xi} \equiv 1.0$). The nonparametric estimators have optimal $p = 0.12$ in the 'exponential' case and $p = 0.16$ in the 'power' case. Model (5.1) shows very weak dependence of the fit ψ given by (2.1) on the window bandwidth and the corresponding parametric estimator practically coincides with the nonparametric estimators (cf. the better fit on Fig. 1). This suggests that d.e. describing model for VOC emission should include nonlinear expressions of $c(x)$, and may indicate presence of slow desorption processes resulting from interactions between the VOC particles. Such interactions are adequately described in a model d.e. by $c^2(x)$ and correspond to value $C = 1$ in (5.2).

Figure 2 shows the joint pointwise behavior of parameters A, B, C in cases of small window and 'exponential' model, and for both small and large windows and 'power' model. The nine graphs show their rather small stochastic fluctuations and display the amount of numerical instability. It seems to be caused by a tolerant but speeding up the calculations stopping rule in finding minimum of (1.2) and by a flat graph of M near the minimum.

Whenever a parametric model $g(x, \theta)$ is acceptable function $\psi(p)$ given by (2.1) and the value of p minimizing it provide an information on how many observations are necessary to estimate adequately the parameters of the model. In the present case model (5.2) seems to meet this requirement and also suggests changes in the full model of the VOC desorption.

6. References

- Cleveland, W. S. (1979), "Robust Locally Weighted Regression and Smoothing Scatterplot", *Journal of the American Statistical Association*, **74**, 829-836.
- Dunn, J. E. and Chen, T. (1992), "Critical evaluation of the Diffusion Hypothesis in the Theory of VOC Sources and Sinks". *ASTM Symposium on Modeling of Indoor Air Quality and Exposure*.
- Eubank, R. L. and Spiegelman, C. H. (1990), "Testing the Goodness of Fit of a Linear Model Via Nonparametric Regression Techniques", *Journal of the American Statistical Association*, **74**, 829-836.
- Fan, J. (1991), "Local Linear Regression Smoothers and their Minimax Efficiencies", submitted for publication.
- Greblicki, W., Krzyżak, A., and Pawlak, M. (1984), "Distribution-free pointwise consistency of kernel regression estimates". *The Annals of Statistics*, **12**, 1570-1575.
- Hall, P. and Jones, M. C. (1990), "Adaptive M-estimators in nonparametric regression", *The Annals of Statistics*, **18**, 1712-1728.
- Härdle, W., Janssen, P., and Serfling, R. (1988), "Strong Uniform Consistency Rates for Estimators of Conditional Functionals", *The Annals of Statistics*, **16**, 1428-1449.
- Huber, P. J. (1967), "The behavior of maximum likelihood estimates under nonstandard conditions", In *Proc. Fifth Berkeley Sympos. Math. Statist. Probab.* **1**, 221-233. Univ. of California Press, Berkeley.
- Kozek, A. S. (1984), "Influence curve for minimum distance estimators and supremum metrics", *Statistics & Decisions*, Supplement Issue No 1, 131-158.
- Kozek, A. S. (1990), "A nonparametric test of fit of a linear model", *Commun. Statist.-Theory Meth.*, **19**, 169-179.
- Kozek, A. S. (1991), "A nonparametric test of fit of a parametric model", *Journal of Multivariate Analysis* **37**, 66-75.
- Kozek, A. S. and Schuster, E. F. (1990), "Optimal quantile principle for selecting variable bandwidth in regression estimators", *Proceedings of the Computer Science and Statistics: 22nd Annual Symposium on the Interface*, ed. R. LePage, pp. 401-405.
- Kozek, A.S. and Schuster E.F. (1991), "On 'Fit the Short Curve' principle for smoothing nonparametric estimators" *Proceedings of the Computer Science and Statistics: 23rd Annual Symposium on the Interface*, pp. 196-199.
- Macauley, F. R. (1931), "The Smoothing of time series", New York: *National Bureau of Economic Res.*
- Parzen, E. (1990), "Unification of Statistical Methods for Continuous and Discrete Data", *Proceedings of the Computer Science and Statistics: 22nd Annual Symposium on the Interface*, ed. R. LePage, pp. 235-242.
- Pollard, D. (1984), *Convergence of Stochastic Processes*. New York, NY: Springer-Verlag.
- Stone, C. J. (1977), "Consistent nonparametric regression", *The Annals of Statistics*, **5**, 595-605.

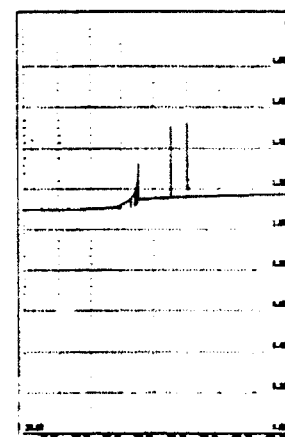
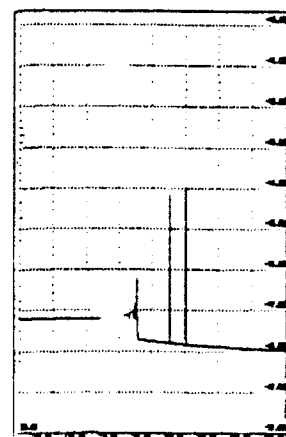
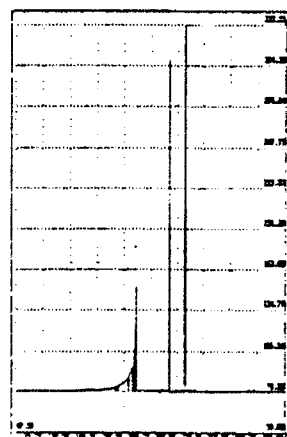
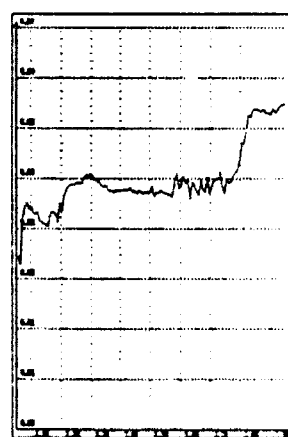
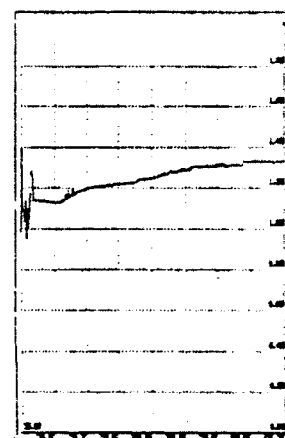
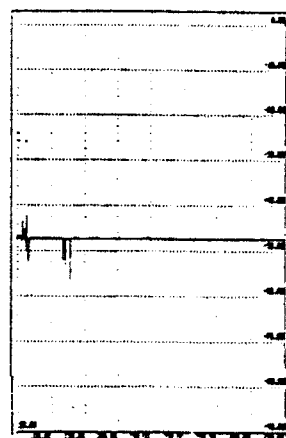
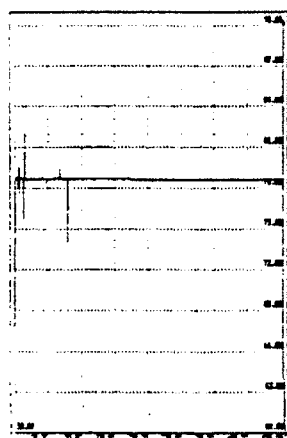
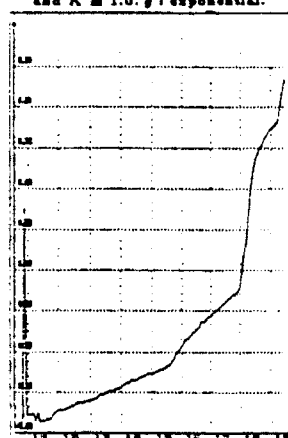
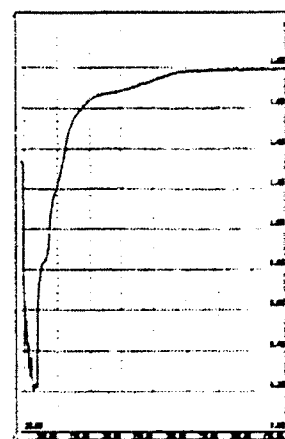
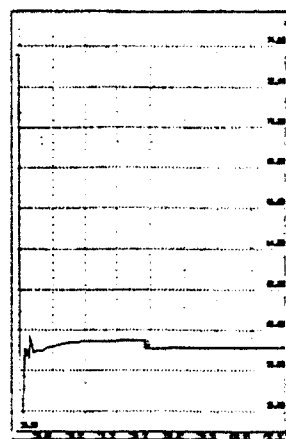
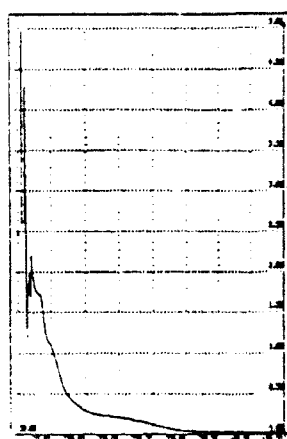
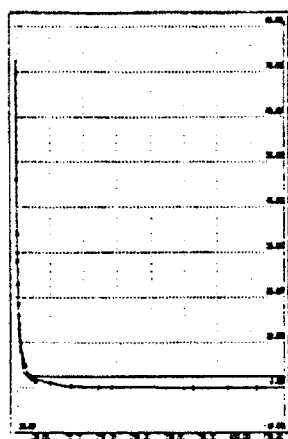


Figure 1.

Figure 2.

AN EXPERIMENT TO INVESTIGATE THE EFFECTS OF REDUCED DATA INK IN VERTICAL BAR CHARTS

Mark Eakin, University of Texas at Arlington

Abstract - Tufte in his book *The Visual Display of Quantitative Information* supports reducing redundant data-ink in graphs. This study reports an experiment to compare vertical bar charts with and without redundant data-ink. The bar graphs without redundant data-ink resemble the types described in Tufte's book. The experiment examines the influence of unnecessary data-ink on a subject's estimated ratio of the lengths of two bars.

INTRODUCTION

Tufte (1983) supported increasing the data to ink ratio in a graph by removing all redundant data ink. An example of redundant data ink would be the widths of bars in a bar graph if only the height had information value. Cleveland (1984) introduced the dot chart that follows Tufte's recommendations by reducing the redundant dimension in a bar chart to a series of dots.

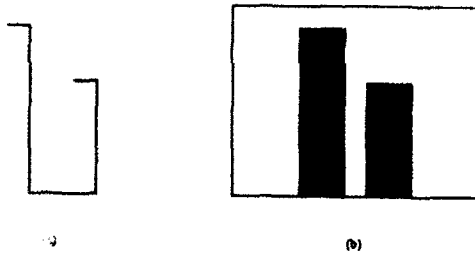


Figure 1. Example Bar Graphs. Example of (a) reduced data ink and (b) regular bar chart as shown to the students during experiment.

However, the results of recent experiments, have questioned the need to remove dimensions that have no information content. Eakin and McKinney (1990) found no significant difference between dot and bar charts for estimating ratios of two lengths. Spence (1990) found that ornamental extra dimensions (bars and solids) do not decrease the accuracy of estimating the ratio of two lengths and may be faster than line segments.

This paper reports the results of an experiment that explicitly tests a form of a bar chart suggested by Tufte. The experiment compared Tufte's form, a reduced bar chart, Figure 1(a), to a regular bar chart, Figure 1(b). No significant difference in estimating the ratios of two lengths were found between the two bar chart types.

EXPERIMENTAL METHOD

The experiment used a cross-over design and was conducted in a computer laboratory. The following discussion of the experimental design includes the subjects, the lab & software, the task, the factors, the protocol, the measured and created dependent variables.

MBA students from an advanced statistics class volunteered for the experiment in return for free group tutoring in statistics. The majority of the students were familiar with bar charts and PCs.

The lab consisted of 20 386 SX computers. Each machine was running a program in compiled Quick basic that depicted the lengths on high resolution VGA color graphics.

After viewing two lengths depicted on the monitor, the task involved asking the subjects to enter their best estimate of the ratio of the larger to the smaller length. Twenty-five pairs of lengths were generated and displayed as both a regular and reduced bar chart. The order of depiction of each pair was randomly sorted within each type of bar chart. The program recorded both the error in the subject's guess and the time that it took for the student to respond with an answer.

The independent variables consisted of four experimental factors and one blocking variable. The factors consisted of the graph type, the length of time the chart was displayed on the screen (unlimited or one second) and two presentation order factors. Because all graphs were seen first with either unlimited or one-second display times, the first presentation factor recorded the sequence of presentation: either unlimited graphs followed by one-second graphs or visa-versa. The second order factor recorded the set of graphs that a subject saw first and the set saw second; this factor attempts to capture any learning effect that might exist. Recording subjects' student identification numbers (IDs) allowed control of variability due to subject's age, experience, graphical perception ability, etc.

All subjects were shown the procedure in the same session. Correct and incorrect responses were demonstrated and every subject was trained on five displays of each graph type for both display lengths. Due to the limited number of computers, the subjects were then assigned into two groups. In each group, half the subjects were assigned to each sequence of display lengths. The

students responses were recorded automatically by the computer.

The variables recored were the time for the subject to respond and the absolute difference (error) between their guess of the ratio and the actual ratio. From these responses, the dependent variables used in the analysis were the median and mean error per subject, the mean and median time to respond per subject, and the natural logarithm of the mean and median error.

ANALYSIS

The data was analyzed using an anlysis of a cross-over design and the results given in Table 1 and graphically in Figure 3. As expected, there is a significant amount of variability due to differences in students. The group, the sequence, and the time to respond did not significantly affect either the mean or median error or its logarithm. However significant differences were found when using median responses that were not found using mean responses and visa-versa. This difference in patterns of significance could be due to the skewness of the error values.

Table 1. P-Values. The analysis of variance was performed using either the mean or median error response per subject and using both error and the natural logarithm of error.

Factor	(a) Mean		(b) Median	
	Err	Logerr	Err	Logerr
Group	.197	.145	.287	.192
Sequence	.208	.329	.548	.483
Student	.001	.001	.001	.001
Display Length	.010	.007	.185	.128
Order	.182	.118	.040	.087
Graph	.462	.762	.047	.043
Response Time	.388	.572	.883	.840

The variable of interest, Graph, showed no significant difference using mean reponses but were significantly different using medians. The mean of the median error for regular bar charts was .166 while the reduced bar charts had .189 error; the minimum significant difference was .0225 with a significance level of 0.05. When the mean of the mean errors were analyzed, the regular bar chart showed .558 error to .582 error for reduced bar charts; the minimum significant difference was 0.066. The increase in mean error going from medians to means per subject indicate how skewed the responses were.

Even though the learning effect was only significant for median response, Figure 3 appears to show a learning

effect from the first to the second set of graphs shown to the subject. The difference in graph types appear to decrease from the first to the second set. While this appears to indicate interaction, no significant interactions were found.

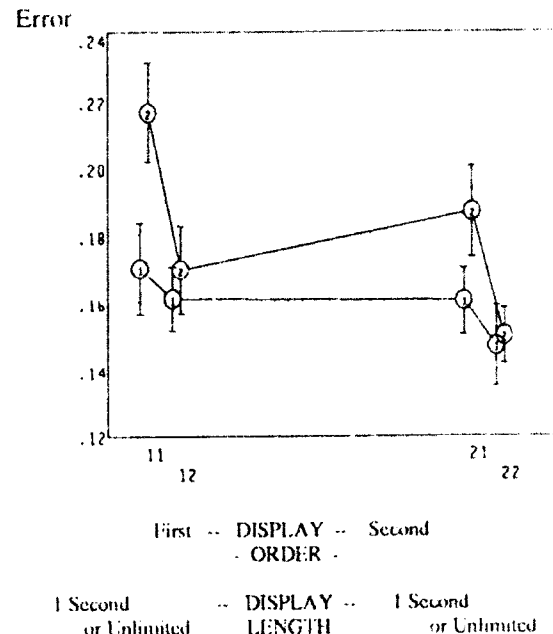


Figure 2. One Standard Error Bar Chart. These values have been corrected for the group, the sequence of presentation and the subject. The reduced bar chart produced the top four connected means with the bottom four belonging to the regular bar chart. The four means on the left belong to the first set of graphs presented to the student and the the right belong to the second set. The first two means on the left set of four occurred if the graphs were presented for only 1 second while the second two on the left were for unlimited presentation time. Similarly this is true for the second set of four.

5. CONCLUSIONS

The results of the experiment tend to support the results found by Eakin and McKinney and Spence. For estimating the ratios of two quantities, the choice of regular or reduced data ink bar charts does not greatly affect accuracy. Any differences found appear to become less with practice.

BIBLIOGRAPHY

Cleveland, William (1984), "Graphical Methods: Full Scale Breaks, Dot Charts, and Multibased

Logging", The American Statistician, Vol. 38, p. 531-534.

Eakin, Mark and Vicki McKinney (1990), "Experiments Comparing Dot Charts and Bar Charts", Proceedings of the Section on Statistical Graphics 1990 Annual Meetings of the American Statistical Association, p. 66-70.

Spence, Ian (1990), "Visual Psychophysics of Simple Graphical Elements", Journal of Experimental Psychology: Human Perception and Performance, Vol. 16, No. 4, p. 683-692.

Tufte, Edward (1983), The Visual Display of Quantitative Information, Graphic Press, Cheshire, Connecticut, p. 123-137.

Statistical Graphics Laboratory Implementation

Lorrie L. Hoffman^{*}
 Department of Statistics
 University of Central Florida, Orlando, FL 32816

Abstract

The aim of this paper is to explain the motivation, planning, acquisition and installation of the statistical graphics laboratory in the Department of Statistics at the University of Central Florida (UCF). Problems and pitfalls will be discussed. Portions of the paper will address current accomplishments and future plans. The author offers suggestions to others to aid in the design of graphical computer hardware projects.

MOTIVATION

The primary objective for acquiring a graphics complex was to introduce a general curriculum course in Statistical Graphics which would be open to quantitatively oriented undergraduate students. The course itself was to focus on familiarizing the student with a cross-section of graphics hardware and software, and enhance the student's capabilities in collecting and describing data. We also wanted to provide for convenience of locality of the equipment and good connectivity of existing and proposed facilities which allows for conservation of resources for future software expenditures.

PLANNING

A team of professors from the Statistics Department drafted a proposal for NSF funds. The responsibility of the initial hardware design fell to this author due to her previous industry employment as a computer hardware designer. Working closely with an AT&T Value-Added Reseller, a preliminary graphics complex platform was proposed.

Several key issues were resolved at this early stage. We decided to use personal computers (PCs) as workstation/terminals rather than diskless workstations or X-terminals (Coulson, 1991 and McMullen, 1990). Here we sacrificed resolution for functionality. X-terminals are 1024 × 1024 pixels whereas the PCs are 640 × 480 pixels, but PCs can additionally run graphics software packages in stand-alone mode. It was also clear at this point that we should purchase

industry-standard equipment, rather than anything that was vendor proprietary. This would allow the greatest flexibility when it came to enhancing the platform with either new hardware or software. Additionally, we emphasized functionality over performance. We wanted the most capability for the money in spite of the possibility of slow turn-around times at the terminals. And we planned at the leading edge of technology. We were aware that the process of grant endowment, formal equipment reviews and bidding, and general paperwork would create a large time span between this initial planning phase and final implementation. This translated into the act of including system software in the proposal which might still be under development.

The proposal was submitted to NSF in the fall of 1988. We were granted funding in the amount of \$46,500 the summer of 1989 (plus equal matching monies from the University). We transformed the NSF proposal into a formal requirements document which went through numerous channels internal to our university. One of those stops was with our Computer Services division. They suggested adding an uninterrupted power supply (UPS) to protect our equipment from power hits (which are commonplace in lightening-prone Orlando) and a large capacity high-speed tape device for backups. After rewrites, we submitted our equipment needs document to four vendors: IBM, SUN, AT&T and DEC. Each of them arranged presentations of their wares and preliminary price quotes.

ACQUISITION

The formal request was made public by our purchasing department during the spring of 1990 and several vendors made bids. After opening and reviewing these, the team reminded themselves that the platform was built around the idea of getting hardware which could most readily use software which already existed (on mainframes, networks, etc.)—our resource conservation philosophy. DEC's solution offered the greatest flexibility from this standpoint, mostly because UCF has a free site license for all DEC developed software.

^{*}This implementation was made possible by an equipment grant #USE-8951299 from the National Science Foundation.

The signing of the contract was delayed because one of the vendors filed a protest which our purchasing office had to resolve. Further delays ensued when our equipment room modification took longer than anticipated. Finally in the spring of 1991 the DEC hardware installer arrived.

During this time our initial DEC sales representative retired and the interim sales representative was again replaced mid-summer, 1991. This caused some communications problems and lack of continuity on the account which contributed to subsequent troubles with product functionality.

INSTALLATION

The complex itself consists of two minicomputer servers attached to the UPS. One of the servers (supplying the graphics software capabilities) is a DEC 3100 minicomputer which is RISC based (reduced instruction set chip) with 12 MB main memory, a 94 MB cartridge tape drive with 665 MB of SCSI (small computer system interface) hard disk storage with a monochrome terminal. It runs ULTRIX V4.0 and has TCP/IP along with DECnet networking capability. The principal graphics software is SPLUS[®] from Statistical Sciences, Inc. The other server (providing support for networking of the PCs) is a VAX 3100 with 8 MB main memory, a 94 MB cartridge tape drive with two SCSI hard drives of 105 MB and 332 MB storage and a monochrome terminal. It runs VMS V5.0 and has DEC's PATHWORKS-server software which supports TCP/IP and DECnet communication.

The ten 316SX personal computers based on INTEL's 80386SX chip running at 16 Mhz are hooked up via thinwire ethernet to DEC's DEPCA cards. The PCs have 2 MB main memory, 40 MB hard disk and 3 1/2 inch diskette drives with mouse and VGA color monitors. They run on Microsoft's MS/DOS[®] and use DEC's PATHWORKS for DOS to communicate on the complex.

The printing facilities are controlled by the VAX 3100 talking to a DECSERVER200 on a LAT (local area transport). We currently have DEC's LN03 laser printer and their LJ250 dot matrix color printer.

The PCs have locally resident software which loads on boot. A menu selection allows the user to activate the MS/DOS environment and then subsequently to issue a site-tailored request for DECWINDOW usage. This brings up both a window to the VAX 3100 and one to the DEC 3100. The other menu selection readies the user to logon to UCF's mainframe (this connection is accomplished through the VAX 3100 communicating with DEC's DEMSB (an SNA box) via a modem.

ACCOMPLISHMENTS

Implementation of the functionality was tackled in steps. Our goals can be described as: 1. get SPLUS[®] capabilities operational, 2. attach to the UCF mainframe and 3. link to the campuswide NOVELL[®] LAN. Goal 1 has been achieved and will be described below. Goal 2 is currently in system testing and goal 3 is part of our future plans.

A possible scenario for data analysis could be: a student brings data in on a 3 1/2 inch diskette. She brings a PC up in native mode and using ftp (a command available due to DEC's PATHWORKS) transfers the data to an ULTRIX file on the DEC 3100. The student then activates the windows environment and lists the data set on the DEC 3100. She then calls up SPLUS[®] and an X11 (MIT, release 4.0) graphics window. Choosing SPLUS[®] commands like contour or plot, she can visualize the data on screen. The final task is to print a hardcopy of the graph. Unfortunately (see PROBLEMS section) a click on Printgraph does not currently suffice. The default command assumes either that the printer is Postscript[®] compatible or is an HP Laserjet printer. After translation the graph can be printed on the LN 03 or color LJ 250.

PROBLEMS

Our problems can be summarized by the four P's: proprietary, performance, payment and personnel.

We had several problems which stemmed from installing proprietary software. The initial hookup of the networked complex was done using DECnet. Unfortunately, when SPLUS[®] was installed we found ourselves unable to call up an X11 graphics window. This was solved by reconfiguring with TCP/IP packet communication protocol. Additionally, DEC delivered printers which communicated in DEC's sixel graphics language. The problem was circumvented by purchasing a DEC software product called PS-PRINT which translates Postscript[®] files to sixel. Unfortunately, the translation is a painfully slow process.

This leads to the discussion on performance (read "user's response time"). With an ex-computer performance design engineer on the team, we were aware of the potential for disk and cpu bottlenecks, excess memory usage (paging) and network contention. We are working to alleviate these troubles.

In the payment arena, we found the entire ordering/billing/receipt phase to be a nightmare. We were frequently working with the technicians at the last minute to order software which should have been shipped earlier. Plus, DEC's invoice numbers rarely corresponded to those listed in the bid. This was troublesome to our university property office who was trying to tag equipment.

Lastly, ongoing maintenance of a hardware platform of this sort is a personnel sinkhole. Routine maintenance activities such as adding user accounts, doing backups, license updates and software upgrades consume a lot of manpower.

FUTURE PLANS

Our main goals in the near future are to write a user's manual, to investigate public domain software and to finish goal 3 (NOVELL[®] LAN connection).

SUGGESTIONS

To anyone contemplating a computer complex acquisition we offer the following tips: get the vendor to show you a working installation similar to what you need, badger the sales representative for the best deal, get a commitment from your boss for ongoing system administrator support, carefully define objectives and prioritize needed functionality (you probably "can't get it all"), be prepared to be patient. Our installation process extended a full year beyond what we anticipated.

REFERENCES

- Coulson, C. J., "The Performance Road Test," DEC PROFESSIONAL, June, 1991, pp. 50-58.
- McMullen, J., "Have PCs Buried the Graphics Terminal Market," Datamation, June 1, 1990, pp. 69-72.

The POSTSCRIPT Imaging Model

Chen-Chi Shing

Computer Science Department, Radford University, Radford, VA 24142, USA

Abstract

The PostScript imaging model is the set of rules that are incorporated into the design of a graphics system. The PostScript, a device independent page description language is designed to integrate text and graphic images. Any graphics package which can generate a PostScript output file, such as SAS/GRAPH, Mathematica, Maple or Island Graphics running on SUN workstations, can be integrated into a LATEX Postscript file. Unix tools are created to print them on APPLE LaserWriter II or viewed in SUN Openlook.

1. Introduction

The campus of Radford university has been under networking since 1990. Basically, at least one subnet was set up within each department and there is one SUN file server running under SUN/OS 4.1 and serving as the master of the Network Information System in its subnet. Users can share data and software within campus local network and communicate with any other internet users through tcp/ip. Graphics packages are spread within different subnets. For instance, Maple V, MathStation 1.2 and MatLab 3.5 are within math/statistics subnet; SAS 6.03 and SPSS 4.0 are used in computing center subnet; however, Mathematica 1.0, Island Graphics 1.0 and Latex 2.95 were installed within computer science subnet. There is a need to integrate the output of different graphics packages into a file and print them in a PostScript laser printer.

So far there doesn't exist a software in the network which can integrate any other graphics output. Because PostScript is a device independent page description language which can perform graphics capabilities and most graphics packages can generate PostScript output, it is possible to integrate them easily.

This paper will first describe what PostScript language is and how it is used in graphics. Then we will explain how to integrate some graphics output using PostScript. An example of a SAS/GRAPH output combined into a LATEX file will be shown.

2. PostScript

PostScript was designed and developed at Adobe Systems Incorporated by John Warnock, Chuck Geschke, Doug Brotz, Bill Paxton and Ed Taft in 1982. It is a powerful device independent interpretive language and mainly used for graphical printing in two dimensional printed pages. And It is used in software packages rather than by programmers. However, it includes most programming language features and powerful built-in graphics primitives.

2.1. Language

The character set of the language includes all ASCII characters, character space, tab and the newline character. Some characters are used to represent data types. For example, the character % starts for a comment. The character / starts a name literal. The characters (and) enclose a string. The characters { and } enclose an array and the characters { and } enclose a procedure body.

All data accessible to the language, exist in the form of objects. Each object has a type, some attributes, and a value. The complete list of object types is: integer, real, boolean, array, packedarray, string, name, dictionary, operator, file, mark, null, save and fontID. The name of an object is any token containing any number of ASCII characters except special purpose characters described as above. The attributes of an object affect the behavior of the object when it is executed or when certain operations are performed on it. It can be either literal or executable.

While in execution of a PostScript program, the interpreter manages four different stacks. The first kind is operand stack, in which it holds any valid operand object and results of the execution of the PostScript operators. The second kind is dictionary stack. It is the context in which name lookup and definitions occur. The third kind is execution stack, in which it holds only executable objects, such as procedures and files, that are in partial stages of execution. The last kind is graphics state stack, in which it holds all graphics operator after using the gsave operator.

2.2. Graphics

PostScript is a page description language. It allows your software and printer to communicate about the

shape and placement of text and graphics on a page. Basically, it is a graphics package developed to insulate the application writer from the machine dependent details of printers, and to aid in the layout of printed pages in the publishing industry.

PostScript has more than 250 operators. Those operators form six major groups. The first group is graphics state operator that defines the context in which the other graphics operators execute. The second group is coordinate system and matrix operator. The matrix operators can be used to achieve any combinations of translation, scaling, rotation and reflection of any graphics in the user coordinates. The third group is path construction operator. The operators can construct straight and curved line segments to the current path. The fourth group is painting operator. They can place marks on the current page in raster output. It may use a variety of color models to specify output color, halftone screens, and sample images. The fifth group is character and font operator. There are different kinds of Times and Helvetica fonts available. The last group is device setup and output operator. These operators establish the association between raster memory and a physical output device.

The default user coordinate system used in PostScript has the origin at the bottom left corner of a 8.5" by 11" letter. The unit in the coordinate system, point, is 1/72 inch. In other words, $(X_{max}, Y_{max}) = (612, 792)$.

The basic graphics operators can allow users to move to any point in the user coordinate system and draw a line or an arc in the current path. In fact, users can create any forms of objects, such as a circle, an ellipse and a polygon easily based on line and arc objects. It is very easy for users to draw graphics since they can save any graphics state in the stack while working on any other graphics.

2.3. Editor

In any Unix environment one can invoke a text editor *vi* to compose a PostScript file. However, in SUN OpenLook environment there is an interactive PostScript previewer *pageview*. *Pageview* renders a document, one page at a time, onto an offscreen bitmap which may be of arbitrary size, resolution and orientation. The tool brings up a text editor with the PostScript document in it and a window which contains all of the errors and other output from the document. The user may make changes to the document and press the *run* button to rerender the page.

3. Integration of Graphics Package Output

Some graphics packages such as Maple, MatLab and SAS/GRAPH can create generic PostScript output. To integrate them, one needs to first arrange the output layout on paper by changing the coordinate system used in each package and scaling, rotation and reflection if necessary. Then the coordinate system must be changed back to the default user coordinate system before combining the next PostScript file.

Some graphics packages such as Mathematica and Island Graphics create only special PostScript files. Then in order to integrate them, one needs to write utility programs which use Unix scripts *sed* and *awk* to change them to generic PostScript files first.

The following example shows how to combine a SAS/GRAPH into a Latex PostScript output file:

The first step is to allow a SAS program to create a PostScript file for SAS/GRAPH. One needs to enter the following two lines in the SAS program:

```
filename infile 'outfilename';
```

```
and
```

```
goptions device=psepsf gaccess=sasgaedt gsf-  
name=infile gsfmode=replace noprompt;
```

In the *filename* line the temporary file *infile* links the permanent file *outfilename*, where *infile* is the file created by SAS during execution and *outfilename* is the PostScript file created and stored in the current directory after execution. On the other hand, in the *goptions* line the device driver for the PostScript software is *psepsf*. This driver will group the PostScript output as a file. In the mean time *gaccess=sasgaedt* will automatically add a carriage-return and a line feed at the end of each record of the PostScript file.

Since the graphics output created above is a PostScript file which can be edited, we can combine it into a Latex file now. We first change the size of the graph so that it can be fit into the page of the Latex output. Since the origin of the default coordinate system for any SAS/GRAPH PostScript file is set at the upper left corner, we need to first move the origin to the bottom left hand corner (200,2500) before drawing the graph. Then we need to invoke the PostScript commands in the Latex PostScript output *@beginspecial* and *@endspecial* to the SAS/GRAPH PostScript output file which will start drawing the graph. A shell script program using *sed*, *awk* and *tail* commands was created to combine the files.

4. Conclusion

As shown above graphics packages can be integrated through PostScript description language. Because a few foreign characters such as Chinese characters can be drawn by any graphics package, e.g. Island/Draw, those graphics objects can be created in the form of PostScript procedures. They can be called easily in an integrated PostScript program. Finally, a nice report can be generated by integrating graphics output from many different graphics packages and word processor LATEX.

5. References

- Adobe Systems Incorporated, (1985), *PostScript Language Reference Manual*, New York, NY: Addison-Wesley.
- Adobe Systems Incorporated, (1985), *PostScript Language Program Design*, New York, NY: Addison-Wesley.
- Adobe Systems Incorporated, (1985), *PostScript Language Tutorial And Cookbook*, New York, NY: Addison-Wesley.
- SAS Institute Inc., (1988), *SAS/GRAPH USER'S GUIDE*, Release 6.03 Edition, Cary, NC: SAS Institute.

Handwritten Digit Recognition Using Deformable Templates

Alistair Sutherland*

Department of Statistics and Modelling Science
University of Strathclyde
26 Richmond St.
Glasgow
Scotland G1 1XH

Abstract

A new method for recognizing hand-written digits is presented. The method is based on the idea of "deformable templates" originated by Ulf Grenander. Each of the ten digits 0-9 is represented by a stochastic model consisting of a series of linked rectangular segments. The coordinates of the start- and end-points of the segments form a multivariate normal distribution. Provisional results are presented for a dataset of 20,000 handwritten digits gathered from Zip codes on letters.

1. Introduction

The recognition of hand-written characters is a subject of great commercial interest. The most immediate areas of application are the recognition of postcodes [Le Cun, 1990] and figures on checks. Other areas include signature verification. If the technique could be extended to Chinese or Japanese characters it would make easier the design of a computer interface for Chinese or Japanese people in their native language [Gao & Li, 1989].

A survey of early methods in character recognition is given by Mantas [1986]. Most are based on first of all thinning or skeletonizing the character and then classifying each pixel, given its neighbors, as an end-point of a line, a crossing point, a T-junction or a point along a line. This leads to the identification of the various strokes and then to a classification of the character. However, this method is not very robust to noise. Common features of hand-written characters such as gaps in lines, strokes which do not join up, extra flourishes and non-straight lines can lead to mis-identification of the strokes and so to mis-classification of the character. Because of these problems attention has now switched to neural networks as the best hope for improved recognition rates. In this proposal we hope to compare the performance of neural network algorithms with a new technique 'deformable templates'.

We have been given a large dataset of hand-written digits by the German company Daimler-Benz. The dataset consists of 20,000 examples of the digits 0-9 with 2000 examples of each digit. The characters are digitized onto 16x16 pixels with 256 grey-levels per pixel. They have been normalized so that they all have roughly the same size and stroke width. Daimler-Benz's main interest is the recognition of German Zip codes which consist only of digits. To be of any commercial use recognition rates of over 99% are required.

This dataset has been provided as part of the ESPRIT project 'StatLog' [Sutherland & Feng, 1992]. As part of this project many discrimination algorithms have been applied to the dataset, including Fisher's linear discriminant analysis, Rumelhart's backpropagation algorithm [Rumelhart *et al*, 1986], decision tree algorithms such as CART and ID3, Bayesian algorithms, nearest neighbor and kernel-density algorithms. The best performances were as follows

k-nearest neighbor	98%
backpropagation	95%
kernel-density estimation	93%

None of these methods use any background knowledge about the data. For instance, we all have a rough idea of what a '4' should look like. It is possible that an algorithm which did use such background knowledge would achieve a higher recognition rate.

2. Deformable Templates

To make use of this background knowledge we now propose a new algorithm based on the idea of deformable templates. Rigid templates have been used extensively in image analysis but *deformable* templates are much more recent and differ from rigid templates in that they consist of a number of constituent parts which can move relative to one another according to some probability distribution. This idea grew out of work by Grenander

*Funded by European Community under ESPRIT project 5170 "StatLog"

et al [Chow, Grenander & Keenan, 1991, Grenander & Keenan, 1989] who originally applied it to the recognition of images of hands. It was applied to galaxy classification by Ripley and Sutherland [1990] and to segmentation of biological images [Ripley, 1990 and 1992]. It is the galaxy classification work which is most relevant to digit recognition. Galaxies were modeled by chains of rectangular segments whose angles, lengths, widths and brightnesses formed a Markov chain. Samples S were generated from the probability distribution defined by this model and accepted or rejected with probability $P(Z|S)$, where Z is the original data.

We propose to apply a similar idea to the hand-written digit problem. Each digit will be modeled by its own template consisting of a number of rectangular segments. The templates for the digits '4' and '0' are shown in fig.1. Each segment is defined by five parameters: the coordinates of its start- and end-points and its width. In the case of the '4' two of the segments have a common point and so the number of parameters is reduced by two - the total number of parameters is therefore thirteen. In the case of the '0' there are four common points, so the total number of parameters is $5 \times 5 - 8 = 17$. Instead of a Markov random chain we now assume these parameters obey a Multivariate Normal Distribution (MVN).

In this paper we will work with a version of the data, in which the greylevels have been binarized. The templates therefore have a brightness of 1 inside each segment and 0 outside. It is possible to extend the method to the case of 256 grey-levels but at the expense of computational speed.

The covariance matrix of the MVN must be estimated. This is done in the 'training' phase of the algorithm, using a subset of the data. The accuracy of the algorithm is then tested in the 'classification' phase when the algorithm is used to classify new data. The classes predicted by the algorithm are compared to the true classes to calculate the accuracy rate.

2.1. Training Phase

In the training phase the algorithm estimates the covariance matrix of the MVN as follows. We take a subset of the total dataset and treat it as a 'training' set. The training set consists of 500 examples of each of the ten digits. We then try to fit the appropriate template to each digit so that the number of mismatched pixels is a minimum. Simulated annealing is used to minimize the number of mismatched pixels. At each iteration of the simulated annealing procedure a new sample of the template is generated by changing one of the parameters at random. The new value is chosen from a uniform distribution over as large a range as possible, e.g. a

co-ordinate of one of the end-points is distributed over the whole width (16 pixels) of the image. Simulated annealing is used because it avoids being trapped in local minima, unlike other optimisation techniques such as conjugate gradients. It has the disadvantage that it is very slow, some 40,000 iterations are required to reach the minimum in this case. However, the training phase is carried out only once, so speed is not important.

Some of the original digits and the fits achieved by this process are shown in figs.2-3. Note how the algorithm has correctly identified the length, width and orientation of each stroke. The fitted template still has to be inspected visually to check that the right part of the template has been fitted to the right part of the image. However, this is a reasonably quick process. Once we have fitted some large number of digits we can then estimate the covariance matrix for each template.

2.2. Classification Phase

At the moment an *ad hoc* method is used to classify new digits. Each of the ten templates is fitted to the new digit and the one which fits best is chosen as the classification. The fitting process, as in the training phase, again uses simulated annealing to minimize the number of mismatched pixels. But this time, at each iteration, the new sample is generated from the MVN, instead of from a uniform distribution. This requires far fewer iterations to reach the minimum. The problem with this method is that the MVN can generate samples which do not resemble the digit on which the template was based. This allows the template to fit onto digits other than the one which it represents. For example, fig 4 shows how the template for an eight can distort to fit onto a seven, by reducing the size of the lower loop until it virtually disappears. To prevent this, we have to insert constraints into the fitting process. For example, in the case shown in fig.4, we could impose a minimum limit on the diameter of the loops of the eight. At each iteration of the simulated annealing procedure the new sample generated by the MVN is tested to see if it obeys the constraints. If not, it is rejected and a new sample is generated. However, even with such constraints mismatches can still occur, like the one shown in fig.5, where a zero template has fitted onto a digit which is actually a one. Using this method, an accuracy of 94% is achieved.

2.3. Future Improvements

Clearly, we need to achieve a higher accuracy than this. The problem with the above method is that it does not use the prior probabilities of the fitted templates when deciding the class: classification is based only on

the number of mismatched pixels. Therefore, a new digit may be assigned to a class, whose template fits well, despite the fact that the template may have been distorted into a shape which has a low probability according to the MVN. To be properly Bayesian we need to take the prior probabilities into account.

One way to use the prior probabilities, which are given by the MVN, is as follows: let S represent the template and Z represent the data. Then,

$$P(S|Z) \propto P(Z|S)P(S) \quad (2.1)$$

using Bayes Theorem. $P(S)$ is given by the MVN but estimating $P(Z|S)$ is more difficult. It is reasonable to suppose that $P(Z|S)$ depends on the number of mismatched pixels between Z and S . If we assume that the probability, p , for any given pixel of being mismatched is the same for all pixels and independent of the other pixels then

$$P(Z|S) = p^N(1-p)^{256-N} \quad (2.2)$$

where N is the number of mismatched pixels. However, the above assumptions are unlikely to be true in practice, and so some work remains to be done on finding a more realistic expression for $P(Z|S)$.

Assuming we can find such an expression, the objective would then be to maximize $P(S|Z)$ for each of the ten templates, i.e. to find the most likely template S given the data Z . We would then compare the maximum values of $P(S|Z)$ for each of the ten templates and pick the maximum of the ten as the classification.

3. Conclusions

Deformable templates have not yet surpassed the other discrimination algorithms in accuracy. However, they have not yet made full use of the prior probability distribution, and this should increase accuracy. The main advantage of the deformable template algorithm is that it allows human knowledge to be built into the algorithm. Whereas kernel-density estimation and neural networks start off with no knowledge of the data, we can give deformable templates our own knowledge about what each digit looks like. This restricts the search space and should make learning more robust and accurate. A second advantage is that deformable templates can give an explanation of why it classifies a given example in a given way - it can indicate which strokes it thinks are which. This also allows human intervention during training - if the algorithm fits part of the template to the wrong part of the digit the human user can adapt the template model so that this does not happen.

4. References

- Chow, Y., Grenander, U. & Keenan, D.M. (1991) *HANDS. A pattern theoretic study of biological shapes*. Research Notes in Neural Computing, Vol 2, Springer Verlag.
- Gao, Q. & Li, M. (1989) "The minimum description length principle and its application to online learning of handprinted characters" *Proceedings of the 11th International Joint Conference on Artificial Intelligence Vol 1.*, 843-848
- Grenander, U. & Keenan, D.M. (1989) "Towards automated image understanding" *J. Appl. Statistics* 16, 207-221
- Le Cun, Y. et al (1990) "Handwritten digit recognition with a back-propagation network" in *Advances in Neural Information Processing Systems 2* Ed. Touretzky, D.S., Morgan Kaufman, San Mateo, 396-404
- Mantas, J. (1986) "An overview of character recognition methodologies" *Pattern Recognition* 19, 425-430
- Ripley, B.D. (1990) "Recognising organisms from their shapes - a case study in image analysis" in *Proc. XVth International Biometrics Conference*, Budapest, 259-263
- Ripley, B.D. (1992) "Classification and Clustering in Spatial and Image Data", *Analyzing and Modeling Data and Knowledge, Proceedings of the 15. Jahrestagung von Gesellschaft fuer Klassifikation*, Salzburg, Ed. Schader, M., Springer-Verlag.
- Ripley, B.D. & Sutherland, A.I. (1990) "Finding spiral structures in images of galaxies." *Phil. Trans. Roy. Soc. A* 332, 477-485
- Rumelhart, D.E., Hinton, G.E. & Williams, R.J. (1986) "Learning Representations by Back-propagating errors." *Nature* 323, 533-536
- Sutherland, A.I. & Cao Feng (1992) "StatLog - an ESPRIT project for the comparison of statistical and logical algorithms" *Seminar on New Techniques and Technologies for Statistics*, Bonn, February 1992.

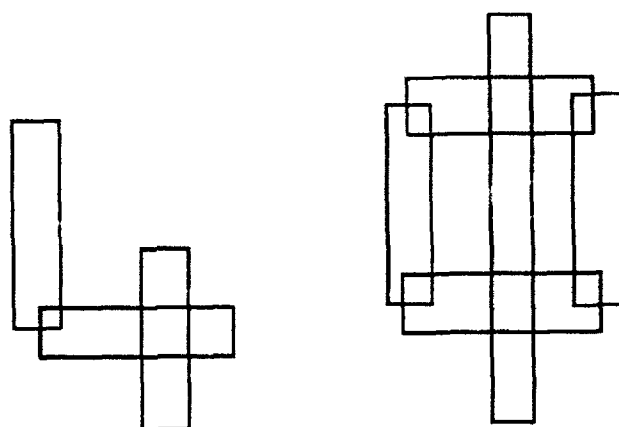


Figure 1: *Examples of templates for the digits 4 and 0.*

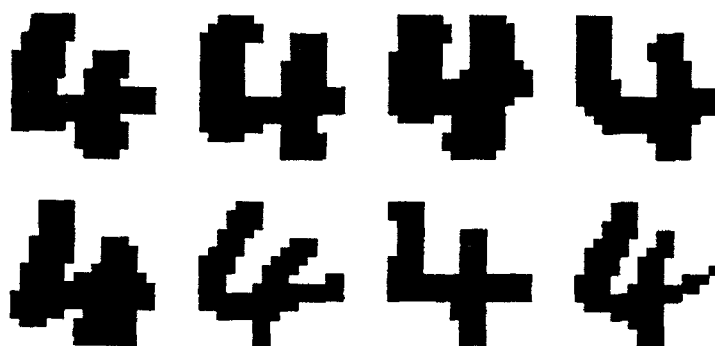


Figure 2: *Some examples of original handwritten fours.*

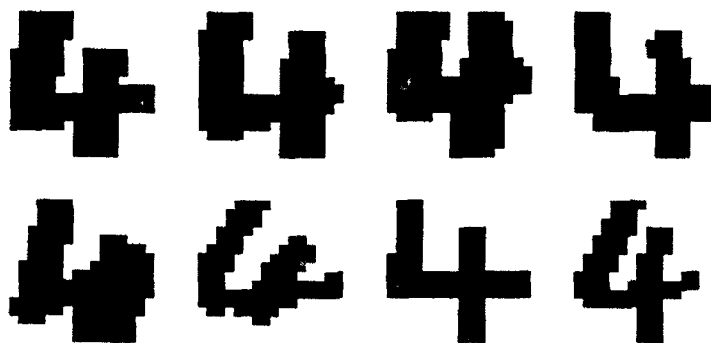


Figure 3: *Templates fitted to the digits in fig. 2 .*



Figure 4: A template for an eight (left) which has fitted onto a seven (right) .



Figure 5: A template for a zero (left) which has fitted onto a one (right) .

Spatial, Statistical, and Graphical Dimensions of Data Quality

M. Kate Beard

NCGIA, Department of Surveying Engineering
University of Maine, Orono, ME 04455
email BEARD@MECAN1.BITNET

and

Barbara P. Battenfield

NCGIA, Department of Geography
SUNY-Buffalo, Buffalo, New York, 14261
email GEOBABS@UBVMS.BITNET

Abstract

The meaning of data quality is associated with fitness for use. The life span and multiple use of data within a Geographic Information System (GIS) generally requires that quality be assessed repeatedly and from the varying perspectives mandated by varying types of analysis. Collection and maintenance of data quality information has been a goal for producers of digital data, however its incorporation into digital databases only recently has become a priority. The well-known advantage of GIS analysis is that data can be composited in both numerical and visual fashion, to facilitate interpretation of patterns of spatial autocorrelation, covariance, and similar measures. The intention of integrating data quality with data is to provide numerical and visual interpretation of certainty and reliability of interpreted pattern.

The credibility of statistical inference and spatial reasoning using GIS may depend on the incorporation of quality information within the database and the display. The disadvantages of visual tools in analysis relate to bias and distortion that may occur in interpreting illogical or poorly designed graphical displays. This paper reports on a research initiative on visualization of spatial data quality. The primary goal of the initiative is to explore, adapt, and evaluate visualization techniques for the representation and communication of spatial data quality. The research builds on traditional cartographic methods combined with visualization techniques made possible by new technologies. The paper begins with a discussion on the nature of spatial data quality which borrows from the definition of quality incorporated in the current federal Spatial Data Transfer Standard. From this discussion the paper identifies problems and research directions needed to build the theory and methodologies for the visual display of data quality for spatial analysis. The structure of the discussion is based around one of the research initiatives of the National Center for Geographic Information and Analysis (NCGIA).

1.0 Introduction

New hardware and software currently allow fast processing and display of large volumes of information. National spatial databases are becoming available, and regional and local databases are beginning to accumulate. These databases are becoming accessible to large numbers of people to support a wide range of applications. With the development and dissemination of these databases, users of spatial data become increasingly removed from the details of data collection and processing. Historically, data collection and preparation were the basis for an awareness and understanding of data quality. As databases become distributed and shared by multiple users, this intimacy with the data and an understanding of its limitations are lost. Computer generated maps, the standard output of a GIS, offer no clues and generally imply an accuracy not warranted by the data. Misinterpretation of GIS results can lead to false claims, poor judgments and even litigation (Epstein and Roitman 1987). There is a sense of urgency within the GIS community to address these problems by developing capabilities to accumulate, store, and communicate metadata, that is the information describing the data in the database (see Lanter and Veregin, 1990).

Information on the quality of spatial data and databases has formed a major concern for both developers and users of GIS (Chrisman 1983). Producers are concerned about the utility and credibility of their products, and users need to be concerned about the reliability of interpretations and decisions which can be made from such products. The quality of spatial information and spatial information products is multidimensional and complex. The quality of information varies spatially and temporally, and the need for such information will vary by application. If one assumes that data have been processed and checked sufficiently that gross errors have been removed, the problem of informing users of the appropriateness of data for their needs must still be addressed.

The volume of information required to adequately describe spatial data quality is potentially quite large, and communicating this pool of information to users presents a

challenge for GIS development. Visualization has recently been proposed as a technique for making complex information more comprehensible. It has been variously described as:

- the organization of abstract concepts into meaningful pictures;
- the transformation of numerical data into understandable images; and
- the manipulation of geometry, color, and motion.

Although visualization has been the essence of cartography, the employment of visualization tools for the representation of quality have not been well developed or are not available in existing GIS packages. Visualization is an efficient means of communicating spatial representations and multivariate data and is thus a potentially effective method for capturing, interpreting, and communicating quality information to users of GIS.

This paper reports on a research initiative currently being conducted at NCGIA sites. The primary goal of the initiative is to pursue research to explore, adapt, and evaluate visualization techniques for the representation and communication of spatial data quality. The initiative began a year ago, and will continue throughout the coming year, with research projects under investigation in a number of disciplines, including GIS, spatial statistics, computer science, and psychology. The highest priorities cited for initiative research include development of conceptual frameworks that link spatial data quality components with visualization methods, prototype development of specific visualization techniques, database management of quality information in support of visual representation, and evaluation of visualization techniques in the context of user needs and perceptual and cognitive skills. Each of these general research areas are discussed in the sections which follow. They include some examples of research priorities and of projects currently under investigation. As background for this discussion, however, we first present a description of spatial data quality.

2.0 Data Quality Components

In order to visualize data quality, we must first understand what it is and then determine what aspects can be effectively visualized. One of the most commonly cited components of data quality is error. Commonly recognized errors include those associated with data collection (source error) and the processing of data (process error). Process errors have proven difficult to analyze in many cases, for example in studies of digitizing error, or in modeling error associated with soil mapping (Fisher 1991). In statistics, the concept of least squares error has been applied to determine reliability or 'confidence' in hypothesis testing. A third error component (use error) defined by Beard (1989) is associated with the appropriate application of data or data products.

It is curious that error is often incorporated in statistical models yet there is no analysis of error in most statistical packages in current use. Sensitivity to error varies with the type of data and use. One tends to look for error less in census data because it is the best data provided. One tends to look more for error in a soil map, whose quality can usually be improved with additional money and sampling resources. The understanding of spatial data quality has recently been extended beyond assessment of error by The Proposed Standard for Digital Cartographic Data Quality (Moellering, 1988). This standard introduced a broad framework for evaluating data quality and has been refined in subsequent years in the Spatial Data Transfer Standard. This standard is currently under review for acceptance by federal agencies producing digital data, including the US Bureau of the Census, the US Geological Survey, and the Defense Mapping Agency. The standard includes measures of accuracy (positional and attribute accuracy), consistency, completeness, and lineage. Temporal data quality is subsumed in the Transfer Standard under lineage.

Positional accuracy generally refers to geodetic, surveying, or mapping errors, measured with respect to horizontal and vertical position within a geographic coordinate system (latitude-longitude) or a statistical coordinate system, for example the multidimensional axes generated by principal components factor scores. Attribute accuracy refers to errors associated with thematic variables, which may be due to categorization, classification, or interpolation. Completeness relates to issues of missing data and methods of compensating for missing values. Logical consistency applies to the data structure used to organize information in question. In standard database terminology, logical consistency assures that there are no contradictory facts in the database. In GIS, spatial consistency typically refers to compliance with a set of topological rules (an internal consistency) and attribute consistency, a compliance with a given set of feature codes or expected relationships between codes (an external consistency). Consistency can be inferred from redundant measurements, from past data, or by compliance with a set of expected relationships. For example the logical consistency of a topographic map might measure whether contour lines cross and whether they follow in logical sequence. Lineage tracks information on data sources and processing methods that may have been applied; it is more a chronological record than an indication of error per se.

Sinton (1978) models theme, location and time as interdependent observation required for any spatial analysis. Sinton observes that in data collection one of these components is typically fixed, another is controlled, and the third is measured. For example in mapping Census data, the controlled component (location within enumeration districts) may be associated with errors due to aggregation levels; and the fixed (time) component of Census enumeration may distort the currency of the representation. These errors differ

from the error associated with the measured attribute or theme (e.g. age, income level, or ethnic background), which may incorporate both data collection (source) error and (process) error inherent in the classification algorithms applied to the data.

A framework for a more systematic assessment of data quality components can be built from the combination of Sinton's observation requirements and the proposed standard components. Figure 1 illustrates this framework. Within this framework resolution has been substituted for completeness. The matrix implies that for any geographic observation we potentially have locational, thematic, and temporal components that can be assessed on different levels of accuracy, resolution, and consistency. Lineage would include the respective narrative for each component.

	Location	Theme	Time
Accuracy			
Resolution			
Consistency			
Lineage			

Figure 1. Components of Spatial Data Quality.

Each cell of the matrix in Figure 1 may include specific measures or indices. For example a measure of locational accuracy could be root mean square error (RMS). Locational resolution could be the 30 meter resolution of Landsat Thematic Mapper data or alternatively a spatial collection unit such as a census block, or tract, a town or a county. A measure of locational consistency could indicate compliance with a prescribed set of rules, for example the consistency of topological rules that require 0-cells to bound 1-cells, and 1-cells to bound 2 cells, etc. Measures of thematic accuracy could include the variance or standard deviation of a set of observations. For example it could be a measure of the probability that a soil map unit class is the same as the soil identified on the ground, or that a classified pixel is representative of a class on the ground. A measure of thematic resolution would indicate the units of measurement. For nominal or ordinal data this would refer to the number of classes or ranks. Thematic consistency would assess compliance of thematic relationships with a set of rules. For example the expectation would be that all links of the Interstate network would have consistent major and minor USGS DLG codes. A measure of temporal consistency would assure that a collection of temporal facts were consistent. For example an inconsistency would exist in the following three facts: Event 1 occurred in the interval $[t_1, t_2]$, Event 2 occurred in the interval $[t_4, t_6]$, Event 1 and Event 2 overlap.

Quality reports should include as much detail as possible about the data, but the question is how much of this information do users really need and whether they prefer aggregate measures over disaggregate ones. The framework in Figure 1 decomposes quality into several separate components, but aggregate indices may be preferable on occasion. The obvious problem with aggregate indices is formalizing how measures of quality aggregate, and in what situations the aggregate measure is statistically as well as conceptually meaningful.

3.0 Research Priorities

This section addresses in greater detail the research priorities identified for an investigation of visualization of spatial data quality. The objective is to develop and assess visualization methods which can realistically communicate the accuracy and reliability of spatial data.

3.1 Conceptual Frameworks

The development of conceptual frameworks for visualizing data quality follows logically from the discussion of quality components. The important question is how do individual or aggregate measures of quality map to visual representations. The specific research focus lies in matching data quality components with a specific graphical symbolization method, that is, 'choosing the right tool for the job.' Bertin (1983) formalized the mapping of visual variables (size, shape, texture, color, orientation) to map data through the scales of measurement (Stevens 1946). A conceptual framework for visualizing data quality can be built on the same paradigm. Formal specification of a quality component, for example, positional accuracy measured as a ratio scale variable, can be mapped to a visual variable exhibiting the same (ratio scale) behavior (Clapham and Beard 1991). Research in this area is linked to achieving an intuitive representation of quality that can be easily grasped by users of spatial data.

3.2 Software and Prototype Development

The second major research focus covers the design and implementation of specific visualization techniques for data quality. In exploring design decisions several additional research questions arise. These include issues of scale, contextual dependency of quality, the dynamics of data quality, the relationships among various quality components and maintaining the link between data representation and quality representation. Each of these sub-topics is explored in greater detail below.

3.2.1 Issues of Scale

All geographic or cartographic representations have an associated scale and the quality of data may vary with this scale. In geographic information systems, the representation scale is dynamic and thus any quality components linked to scale may be dynamic. This is particularly true of locational accuracy and resolution. The implication is that visual

representations of quality linked to scale may need to change as scale changes.

A related issue involves the design of visualization techniques to accommodate scale. If the scale of a quality component is quite fine and the size of the geographic area we wish to view is large, a selected geographic scale may be too small to effectively show the quality component. For example if we wish to show positional accuracy using error ellipses and the chosen graphic scale is 1:100,000, error ellipses less than 10 meters will be indecipherable. Additionally the grain of two different quality measures may be orders of magnitude different in which case they can not be effectively shown at the same scale. This raises the issue of whether we can support local and global views simultaneously. Fournier et al. (1982) state that often visualization techniques model objects at predetermined fixed scales regardless of their suitability for different viewing distances. Thus very fine visual detail may be indecipherable at one extreme (small scales) and large featureless areas result at the opposite extreme (large scales). Ideally the level of detail provided by the visualization technique should be linked and or varied appropriately with the scale of the quality measure which is to be displayed. In some instances the visual display could be used to indicate the gravity of the problem relative to scale. For example if a quality component were significant at a particular scale it would be visible and if it were not significant it would be invisible.

3.2.2 Contextual Dependency of Quality

The rigor and level of detail with which quality information needs to be examined varies with the application or task. Too much quality information or computationally expensive visualization techniques may not be cost effective or appropriate for certain tasks. If we consider three different tasks

- digital data production
- spatial decision-making
- scientific research

we can envisage three quite different sets of visualization tools to support each task. In the case of digital data production, the visualization techniques may be designed to support a very narrowly defined set of quality control parameters for a specific data product. If it is a high volume production environment, visualization methods will need to be simple and efficient. For example, quality control for a product may be based on several fixed thresholds for positional accuracy attribute code accuracy and consistency. The visualization techniques might be a simple color scheme which would immediately indicate compliance or non-compliance with the product thresholds. In the spatial decision making context we could imagine a breadth of applications from land fill siting to emergency response. The requirements for quality information are respectively broad and variable. For example in responding to a fire, positional accuracy may be irrelevant but attribute accuracy

on the age and structure of buildings, the presence of hazardous contents, and currency of this information may be critical. In the context of spatial decision-making the requirements for data quality information are potentially the less rigorous and the most varied. The requirements are not simple, consistent and pre-defined as in the case of digital data production. In the case of scientific research we would expect the most rigorous requirements for assessment of data quality. In the proof or disproof of some theory, the quality of the data must be rigorously established and tracked. Quantitative assessments of quality measures would likely be required in addition to qualitative assessments and thus visualization techniques supportive of quantitative analysis would be required. The design of the visualization tools in each case should be tailored to requirements of an application for quality information.

3.2.3 Dynamics of Data Quality

In the context of geographic information systems, each GIS process applied to the data can potentially change the quality parameters of the data. Tracking the propagation of errors through GIS analysis is an important function which can potentially be supported with visual aids. Visual support could potentially require the generation of a new graphic representation after each process and comparison against the preceding ones. An alternative would be a graphic highlighting the changes produced by each process. The design requirements for visual representations would be that the changes be easy to detect and compare. One possible method currently being explored is the use of reference grids shown in Figure 2.

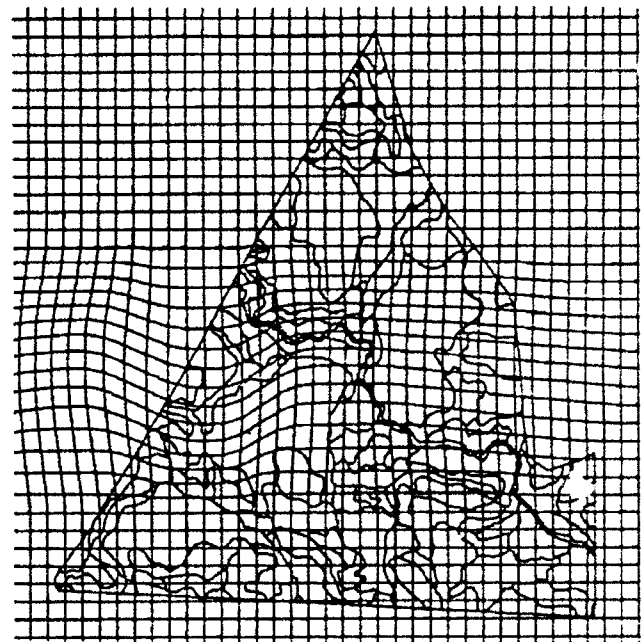


Figure 2. Example of a reference grid as a visual technique to detect and compare change in data quality produced by GIS operations. The grid in this case indicates the nature of geometric distortions produced by rubbersheeting.

The reference grid acts as a backdrop to the data. In one approach a GIS process (transformation) applied to the data is also applied to the grid. Changes which are not necessarily detectable in the data may be detected in the grid, since the expected uniformity of the grid provides a reference for easy detection and comparison of changes.

3.2.4 Relationships Among Data Quality Components

There are certain advantages to decomposing data quality into separate components as was described in Section 2. This decomposition allows users to isolate the components of primary concern and to track them separately. However in many cases these quality components are not independent and an understanding of the relationships between them becomes important. In this case the research questions include: What are the relationships between quality components (i.e. locational and thematic accuracy) and can visual techniques be used to help understand and graphically illustrate these relationships? Similarly can measures of these quality components be combined and can these composite measures be expressed visually?

3.2.5 Linkage of Data Representation and Quality Representation.

Another important design consideration for quality visualization techniques is to assure that there is an unambiguous association between a quality measure and the map component it describes. Quality could be represented as a continuous field underlying an entire map or a discrete measure applying to a discrete object. Once this determination is made, several questions on visual representation arise. In cartographic representations, graphic symbols and visual variables carry data values. In considering design solutions for visualization of quality, choices can be to load the same symbols with quality information, a different set of symbols in the same plane, or to use an entirely different visual mode. Where a line symbol was used to represent the location of a town boundary, color or line type (dotted or dashed) could indicate quality. In this case the link between data and quality is maintained through the same symbol. Alternatively positional accuracy of the boundary could be represented by a band or set of probability bands around the line. In this case a different symbol is employed but the association between data and quality is clear from the spatial relationship in the same plane. The inclusion of quality information in the same plane when the data representation is complex could lead to visual overload and alternate solutions would therefore be required. Linkage of data between different views offers one solution. As an example, data in scatterplots in one window can be linked to the mapped representation of the same data in another window (Monmonier 1989, Haslett et al 1990). Color, highlighting, or flashing provide the visual clues for the user indicating which data are the same in the two views. The advantage of this technique is that the

link between data and quality is maintained and a single view is not overloaded. Additionally, views which are more amenable to quantitative diagnosis than the map view are possible (e.g. a graph).

3.3 Data Models and Data Quality Management Issues

The third research area focuses on data model and database management issues. The objectives here are two-fold:

- to develop data models and structures which explicitly incorporate quality information
- and to develop visual representations which can indicate both the structure of the database and aspects of its quality.

3.3.1 Data Models and Quality

Data presuppose a theory and make assumptions about the real world. Levels of validity and reliability may be implicit in the theory. Spatial data models are developed to support different types of data, but the models can only be as good as the underlying theories. Current spatial theory is not well-attuned to quantification of data quality, especially in categorical data analysis. Consider for example the low reliability of inference associated with soils data underlying current wetlands delineation procedures.

Data quality must be model-based to measure uncertainty. There will always be uncertainty and error at some level of resolution, and users need to be aware of such limitations. For example soils data are collected as measured attributes at a point and transformed to polygon maps. These polygons are not intended to reflect the actual soil type at any one point, but for purposes of management we consider points within the same polygon to be the same type. In this context scale again is an important issue. In moving across scales we often must make inferences about areal data based on the uncertainty of point measurements.

The fitness of data for a particular purpose is clearly important, but assessment is difficult. What is reasonable or unreasonable cannot be determined without a model describing the behavior of the data. The error in raw data and error introduced during data manipulation cannot always be controlled, and for many GIS operations there is no direct way to obtain error measures. A particular statistical problem with spatial data is that replications of GIS models are not typical. Another computational issue centers on the lack of robust models describing error propagation during GIS operations. Without formalized expectations, it is difficult to monitor or predict the generalized behavior of analytical procedures.

Research questions that relate to data models include a diverse range of topics. One question might be posed as

follows: At what level of aggregation (primitive, object, object class, layer, tile, database) should the data quality information be stored and/or linked to the data? For example, the Census Bureau must aggregate some information due to confidentiality requirements. The level at which quality information is stored may depend on the type of measurement, and on what levels of aggregation make sense or are commonly used. For example, a surveyed point or line contains measurement error but the commonly asked question is what is the error in the area of a polygon formed from the point and line measurements. Until a model of aggregation is established and verified, only simple data quality assertions are possible. For triangulated points, where errors tend to cancel, the aggregation model may be driven by the number of redundant observations. Where location is inferred by resampling a satellite image, the aggregation of error will likely not cancel, requiring a different error model.

A second question might be posed as follows: What data models lend themselves to particular representations of data quality? Here the term 'representation' is construed in the general sense, as in digital representation, and not limited to a specific mode such as graphical representation. Dutton (1989) proposed a hierarchical tessellation based on platonic solids in which the tessellation level is indicative of the locational accuracy of the data. Dutton's model offers both a data structure which incorporates quality as well as an effective visual representation of data resolution (see figure 3).

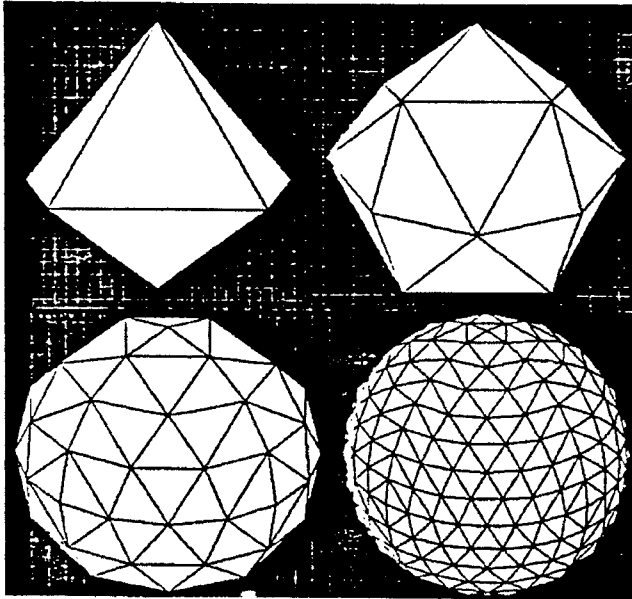


Figure 3. Dutton's quaternary triangular mesh shown to level 3. The model includes both structural and visual representations of quality.

Other hierarchical models similar to the quadtree data structure (Samet 1989) may be useful for some quality

indices. If the quality of information is variable over an area, it should be possible to decompose the area into quadrants until each quadrant has a uniform level of quality. This may provide for automatic system warnings (for example) to users attempting to make inferences about data below the usable limits of resolution, as defined by homogeneity in the data quality hierarchy. One way to link data with metadata in the quadtree structure would be to make data quality and data density interdependent, such that metadata density would vary with data density.

Thirdly, we can ask what types of error models are most effective? Using the temperature of a room as an example, an error model could state that there is a Gaussian standard distribution with an error of 2 degrees. One could generate a map of temperatures within the room, differing only in error. The source of the error differs. The readings could differ in reliability, due to systematic error (which could be adjusted for), or due to the way in which the thermometers were used. The Gaussian model restricts the level of complexity in the model. The Gaussian model describes the probability but gives no information about clustering of data points or sub-areas within the room, as for example near an open window, where errors are likely to be systematic. The process of finding systematic errors can be anecdotal and thus difficult to incorporate into data models even though their existence can be verified. With anecdotal information there is a need to archive metadata in a form that is not readily codified in numeric tabulation or by mathematical formula.

Aggregate measures are complex and statistical models can help. For example, when using attribute variance to predict error, each attribute has a value associated with it. Ideally one would associate an aggregated value with each level of data aggregation. At the Castine meeting, Noel Cressie (Iowa State University) presented the group with an example of a problem whose objective is to map the sudden infant death rate within 100 counties in a contiguous region, and predict the measure of quality. Starting at a low level of aggregation and working up through higher levels of aggregation requires knowledge of covariances between counties, including 100 variance values and a 100 x 100 matrix. Storage of this volume of metadata may not be supported by current storage capabilities. However, it may be most efficient to reduce the covariance information to the computational formula, storing the process to derive metadata if not the actual values.

3.3.2 Management of Data Quality Information

Research questions that relate to data management also include a diverse range of topics. Management of data quality within a GIS database requires attention during manipulation and update, and will likely impact upon future versions of a spatial database.

One of the basic research questions is: Where in the data management process can visualization be used most effectively? It is worthwhile to distinguish between visualization for specific tasks in the life cycle of data product generation and use. Ideally management of quality should begin with data collection and continue over the life span of the data. Visualization techniques could be effectively used at several stages. Visualization techniques could be used during actual data collection. Systems designed for field use could, for example, provide visual clues to indicate where denser sampling might be needed. Such clues could help soil scientists to better capture transition bands. Differing rates of change could be represented by a menu of differing plausible values. Visualization techniques could also be designed to transfer information known by the data collector to the data analyst.

There is also a need for database software designed to examine and manage quality information, for example in query processing. Query languages have only recently been extended to include spatial queries but should also be extended to support queries of quality. For example queries of quality might request: data that is no more than 5 years old, or areas on the map which have positional inaccuracies greater than 10 meters. In most cases the response to such queries will require an appropriate visual representation of the results.

3.4 Evaluation of Visual Solutions

The fourth research area concerns the evaluation of visualization techniques to assure that they match users expectations and their perceptive and cognitive skills. Without a clear sense of what GIS users expect to discover in a data quality display, or even how users comprehend data quality concepts and implications, software and database solutions cannot be optimized.

Visualization and its utility for data quality representation may be viewed from a primarily cognitive as opposed to graphical perspective. From this point of view, one should accept that discovery and innovation have traditionally involved visual thinking. This argues for the importance of understanding information processing capabilities that rely on visual tools and skills. Evaluation of visual solutions to data quality representation requires attention to the internal (perceptual and cognitive) mechanisms by which spatial and temporal patterns are interpreted.

Research questions include:

Can users correctly interpret visual displays of quality?

Are certain visual representations of quality intuitively grasped or will substantial training be required?

Does the incorporation of visual representations of quality within a GIS make a difference? For example, does visual presentation of information about uncertainty alter spatial decision making?

One effort currently underway is to compile a compendium of error or quality classes and associated visualization techniques. These techniques could then be tested by real data users to determine which techniques work best for each error or quality component.

4.0 Summary

This paper has summarized the research agenda formulated by participants in an NCGIA specialist meeting. The research efforts are quite broad and indicate that the visualization of data quality will involve the efforts of several disciplines. It is largely a multi-disciplinary problem which will require input from such fields as cartography, spatial statistics, computer graphics, exploratory data analysis, scientific visualization, psychology, cognitive science, and graphic design. So far research efforts have only touched the surface of the problem. Through publication of the research agenda we hope to enlist the wider efforts of other disciplines in the creation of solutions to this important problem of managing and communicating spatial data quality.

5.0 References

- Beard, M. K. 1989. Use error: The neglected error component. *Proceedings, AUTO-CARTO 9*, Baltimore, Maryland, March 1989: 808-817.
- Bertin, J. 1983. *Semiology of Graphics*. Madison: University of Wisconsin Press.
- Chrisman, N. R. 1983. The role of quality information in the long-term functioning of a geographic information system. *Cartographica* 21 (2/3): 79-87.
- Clapham, S. and M. K. Beard, 1991. The Development of an Initial Framework for the Visualization of Spatial Data Quality. *Proceedings of ACSM*, Baltimore, MD. 2: 73-82.
- Dutton, G. 1989. Modelling Locational Uncertainty via Hierarchical Tessellation. in *Accuracy of Spatial Databases*. Eds. M. Goodchild and S. Gopal. London: Taylor and Francis. 125-140.
- Epstein, E. and H. Roitman. 1987. Liability for Information. *Proceedings of URISA*. 115-125.

- Fisher, P. 1991. Modeling soil map-unit inclusions by Monte Carlo simulation. *International Journal of Geographical Information Systems*, 5/2: 193-208.
- Fournier, A., D. Fussell and L. Carpenter, 1982. Computer Rendering of Stochastic Models. *Communications of ACM*, 25/6: 371-384.
- Haslett, J., G. Wills and A. Unwin, 1990. SPIDER - an Interactive Statistical Tool for the Analysis of Spatially Distributed Data. *International Journal of Geographical Information Systems*, 4/3: 285-296.
- Lanter, D. P. and Veregin, H. 1990. A lineage meta-database program for propagating error in geographic information systems. *Proceedings GIS/LIS '90*, Anaheim, California, November 1990, 1: 144-153.
- Moellering, H. 1988. The proposed standard for digital cartographic data: report of the digital cartographic data standards task force. *The American Cartographer*, 15(1) (entire issue).
- Monmonier, M. 1989. Geographic Brushing: Enhancing Exploratory Analysis of the Scatterplot Matrix. *Geographical Analysis*. 21: 81-84.
- Samet, H. 1989. **The Design and Analysis of Spatial Data Structures**. Reading, MA: Addison Wesley.
- Stevens, S. 1946. On the Theory of Scales of Measurement, *Science*, 103: 677-680.

VISUALIZATION OF FUZZY SCENES AND PROBABILITY FIELDS

Yee Leung

Department of Geography, and
Center for Environmental Studies
Chinese University of Hong Kong
Shatin, NT, Hong Kong

Michael F. Goodchild and Chih-Chang Lin
National Center for Geographic Information and Analysis, and
Department of Geography
University of California, Santa Barbara, CA 93106, USA
good@ncgia.ucsb.edu

Abstract

Fuzzy classification has the potential to yield richer information from remotely sensed images, but there have been few efforts to deal with the issues involved in working with fuzzy classifications in GIS. Analogous data are also obtained when the multinomial classification given to land is treated as mixed, fuzzy or probabilistic. The paper reports on a series of efforts to develop visualization techniques for such data. To support visualization of the inherent variability in such data, and to propagate uncertainty effectively through GIS operations, it is necessary to introduce the concept of an error model as a stochastic process, and to define a method for creating individual realizations of that process.

Introduction

Increasing emphasis on analysis, modeling and decision support within the GIS applications community in recent years has led to a general concern for issues of data quality. If the purpose of spatial data handling is to make maps, then perhaps it is sufficient to require merely that the output map product be as accurate as the input. But the detailed analytic and modeling applications that underlie much of the recent literature of GIS (Tomlin, 1991; Laurini and Thompson, 1992) demand much more stringent and robust approaches. If the input is known to be inaccurate, uncertain or error-prone, then it is important that the effects of such inaccuracies on the output also be known. Without such knowledge, the apparent value of GIS in supporting spatial decision-making may be illusory.

In this paper we take the position that all geographic information is inaccurate to some degree, because it is impossible to represent the continuous variation of the Earth's surface perfectly in the finite, discrete space of a

digital store. We use the term 'accuracy' generically, and assume that it subsumes error from a variety of sources: uncertainty of definition, imperfect replication between observers making subjective judgments, the consequences of mixed pixels in remote sensing, digitizing error *etc.* A spatial database is a representation of geographical reality in digital form, and the output of a GIS process is an estimate of the results of making an equivalent measurement on the ground. In that sense, accuracy in spatial data handling is a measure of the difference between the digital estimate and ground truth. In cases where ground truth is poorly defined, we include variation between observers in this definition of accuracy. Thus a failure of different observers to agree on the class of land cover at a point contributes to the inaccuracy of land cover data.

Although inaccuracy is pervasive in spatial data, some types are clearly less accurate than others. A GPS survey provides known levels of positional accuracy, down to millimeter levels. We focus in this paper on a class of data known to be subject to relatively high levels of uncertainty, and for which there are no such straightforward measures of accuracy. In this class, every point on the plane is characterized by a single value measured on a nominal scale; examples include soil class, land cover class, and land use. We refer to this as a multinomial field. Two data models are commonly used to build digital representations of such fields. The first, the raster model, is used when the field is obtained by remote sensing, by making use of one of a number of standard procedures for classification. In this model, all information on within-pixel variability is lost. The second, or polygon model, partitions the plane into a number of polygons of homogeneous class, thus losing all variability within polygons. The polygon model is also commonly used in mapping multinomial fields, although the boundary

lines on such maps are drawn as continuous curves and need not be discretized to polygons.

Both models are clearly approximations, and although both are in common use, it is rare for the degree of approximation to be made explicit in either case, or for uncertainty to be propagated through GIS processes. In the raster case, fuzzy classifiers provide one way of describing uncertainty, by associating each pixel not with a single class, but with a vector of class memberships, each one interpreted as a measure of belonging. Thus pixel x 's degree of belonging in class i might be denoted by $\pi_i(x)$, and the vector of class memberships might be written:

$$[\pi_1(x), \pi_2(x), \dots, \pi_n(x)]$$

where n is the number of classes.

In the polygon case, inaccuracy occurs in the form of variation within polygons, perhaps at the edges where boundaries are merely approximations to zones of transition (Mark and Csillag, 1989), or perhaps centrally where small inclusions and islands of different classes have not been mapped. Neither of these issues is dealt with effectively by giving the polygon a fuzzy class membership. Instead, it is necessary to abandon the polygon model because it is fundamentally unable to serve as an adequate basis for representing within-polygon variation. Instead, we see the geometry of the polygon model as an artifact of the mapping process, having little value in an effective approach to data quality, and transform to the raster model. Thus both heterogeneity of polygon class and transition near the boundary are represented through the use of pixel class memberships.

While the concept of fuzzy pixel classification is a familiar feature of the remote sensing literature, there has been very little research on the processing of such data within GIS. In part this may be because of concerns over data volume, since n memberships must be stored for each pixel, rather than one integer between 1 and n . In practice, however, it is rare for more than two class memberships to be significantly greater than zero in any one pixel. Fuzzy-classified scenes are difficult to visualize for similar reasons, and it is not clear how measurements such as class area can be made from such data. Thus despite the availability of fuzzy classifiers, and the greater information content of fuzzy-classified scenes, it is tempting to convert such data to a simple maximum likelihood classification on the grounds that the latter are much easier to handle.

The purpose of this paper is to discuss methods of visualization and processing for fuzzy-classified scenes

within GIS. We include with this term not only the results of fuzzy classification in remote sensing, but also derivatives of the polygon model where each pixel is associated with a mixture of classes, or with probabilities of class membership. The next section discusses the meaning of such data from a statistical perspective, and introduces the concept of an error model. The third section discusses a rule-based fuzzy classifier for use in interactive visualization of scenes. This is followed by a description of the environment for visualization of fuzzy-classified scenes developed by the authors. The final summary discusses directions for future research.

Probabilistic Perspective

Consider a raster in which each pixel is associated with a vector of class memberships. The various possible sources and interpretations of this data were discussed in the previous section. To provide a probabilistic interpretation, we assume that the memberships are normalized by pixel:

$$p_i(x) = \pi_i(x) / \sum_k \pi_k(x)$$

Thus $p_i(x)$ is interpreted as the probability that pixel x belongs to class i out of the n classes. This might be interpreted in a mixed pixel context as the proportion of pixel x 's area that is of class i ; or the proportion of interpreters who would have assigned the pixel's area to class i ; or the proportion of pixels with the same spectral response as x that are truly i ; and numerous other interpretations are possible also.

We define the term multinomial probability field (MPF) as a vector field whose value at any point is a normalized vector of class membership probabilities of length n . A raster provides a suitable way of creating an acceptable approximation of such a field in a digital database.

Although a display of pixels showing the membership in each class is informative, it nevertheless fails to convey an impression of uncertainty, suggesting that memberships are expressions of deterministic knowledge, rather than of lack of knowledge, or of fuzziness. A similar situation in geostatistics has recently been the focus of a paper by Englund (1992). When the technique of Kriging is used to create an interpolated surface between sample points of known value, the result is both a surface and a map of uncertainty. In fact the surface is the estimated mean, and the map of uncertainty shows estimated variance around the mean. Englund deviates from common

practice by showing not the map of estimated means, but sample maps from the distribution of possibilities defined by the means and variances. These, rather than the estimated mean, are then used in GIS processing. As a result, Englund is able to provide visually dramatic illustrations of the uncertainty expressed by the estimated variances, but normally ignored in analyses based on estimated means.

Englund's Kriging means and variances provide an error model, or a stochastic process whose outcomes or realizations represent the uncertainty inherent in the data. Goodchild, Sun and Yang (1992) define an error model in the context of spatial databases as "a stochastic process capable of generating distorted versions of the same reality". The best known error model is the Gaussian, used to describe uncertainty in measurements of a simple scalar quantity like the elevation at a point. Each of the outcomes of such an error model provides one possible version of the truth, as it might be interpreted by one soil scientist, or as it might be digitized by one operator.

In the context of an MPF, the probabilities are the equivalent of Kriging means, and a map of them similarly fails to convey an impression of uncertainty. Goodchild, Sun and Yang (1992) describe an error model for an MPF. Each realization is a map in which each pixel is assigned to a single class. Its two essential properties are:

1. between realizations, the proportion of times pixel x is assigned to class i approaches $p_i(x)$ as the number of realizations becomes large; and
2. within realizations, the outcomes in neighboring pixels are correlated, the degree of correlation being controlled by a spatial dependence parameter.

When the spatial dependence parameter is zero, outcomes are independent in each pixel (the case illustrated by Fisher, 1991). However, this is almost certainly unrealistic since few if any real processes are likely to create such independent outcomes. As the parameter increases, outcomes are correlated over longer and longer distances; one suitable interpretation of this is that larger and larger inclusions within polygons are ignored, or fall below the the minimum mapping unit area.

Many commonly used descriptions of map error fail to meet the requirements of an error model, since they fall short of the complete specification of a stochastic process. Such descriptions include the width of an epsilon band, the measures mandated by many map

accuracy standards, the statistics of the misclassification matrix used in remote sensing, and the reliability diagram found on many topographic maps. All of these are useful error descriptors, but fall short of being useful error models. Neither is there a useful connection between many such descriptors and the necessary parameters of error models. For example, it is not possible to connect the parameters in the model described above with such measures as positional accuracy of polygon boundaries, or per-polygon misclassification of attributes.

A Rule-Based Fuzzy Classifier

Uncertainty is endemic to land classification or regionalization, because with few exceptions the Earth's surface is not naturally divided into regions of uniform attributes divided by clear boundary lines. In practice, while some boundaries between classes may follow well-defined lines such as roads, rivers or ridges, other boundaries must be drawn through zones of transition, ecotones, or similarly fuzzy areas. As a consequence, maps of land cover made by different observers may show different boundary positions, and also different numbers of regions and different boundary network topologies. Such uncertainty may be further complicated by imprecision in our language for classification (Leung, 1984, 1985, 1987). Therefore it is essential to have a built-in mechanism for analyzing and displaying uncertainty within a spatial data handling environment.

Conventionally, classification of remotely sensed scenes is performed algorithmically. In supervised classification, techniques such as maximum likelihood (see for example Nilsson, 1965; Duda and Hart, 1973) and the minimum distance method (Wacker and Landgrebe, 1972; Borden *et al.*, 1977; Phillips, 1973) are all procedural. In unsupervised classification, the most common method is cluster analysis (see for example Duda and Hart, 1973; Coleman and Andrews, 1979) which is again algorithmic in structure.

A common drawback of all of these methods is that they cannot handle uncertainty. Fuzzy cluster analysis (see for example Ruspini, 1970, 1973; Bezdek, 1981) and fuzzy graphs (see for example Leung, 1984) can analyze and depict uncertainty in classification in general and image analysis in particular. Nevertheless these methods are mechanical, and cannot communicate to users any knowledge behind the classification.

To make fuzzy classification more flexible, informative and intelligent, a rule-based classifier has been developed within an expert system environment (Leung

and Leung, 1992a,b). In place of an algorithm, the classification scheme is represented by a set of rules indicating how spatial classes are conceptualized and spatial data are classified. A rule in the rule set is generally expressed as:

```
(rule <rule-name>
  if <object 1> <operator 1> <value 1> and/or
    <object 2> <operator 2> <value 2> and/or
    .
    .
    .
  then <object n> is <value n>
  ) certainty is <certainty factor>.
```

The operators can be ordinary inequalities ($>$, $<$, $=$, $>=$, $<=$) or fuzzy inequalities (\geq , \leq , \approx , $\geq\approx$, $\leq\approx$, where \approx means approximately). The certainty factor can be a precise value in some fixed range, a fuzzy number, or a linguistic probability (Gopal and Woodcock, 1992).

In the identification of water from MSS data, a typical rule might read:

If the spectral value in Band 3 (X_3) is *approximately less than 8* and the spectral value in Band 4 (X_4) is *approximately less than 5* then the pixel is a water body, certainty is 1.

Based on evaluations, ground truthing, experts' experience and knowledge gained, rules can be modified, deleted or added according to the rule set without having to rewrite any part of the program, in expert system environments such as those provided by Leung and Leung (1992a,b). The knowledge-based approach is thus more versatile than algorithmic approaches.

Regardless of which approach is used (algorithmic or rule-based), fuzzy spatial classification differs from the non-fuzzy scheme in that it can depict the intrinsic uncertainty of spatial data. Intermediate areas, fuzzy boundaries and fuzzy regions can be identified by gradation, while precise boundaries can be handled within the same framework by coupling high levels of certainty with spatially sharp changes in class memberships. However, to communicate uncertainty to the user, we need to devise an effective scheme for visual display.

Tools for Visualization

In this section we describe the tools we have developed for rule-based fuzzy classification, and visualization of

MPFs. As we argued in the first section, classification procedures are important for remotely sensed imagery, but it is also desirable to be able to visualize MPFs from sources such as land cover maps, in which classification is performed by other means. For this reason, the system is modular in design, and includes a classification module, display module, and modules for data manipulation. It is interactive and uses a graphic user interface, all instructions and operations being triggered by selecting appropriate screen buttons. Windows are opened and closed as appropriate. The system has been developed in C and X Windows for the IBM RS/6000 under the AIX operating system.

Within the classifier module, fuzzy rules are managed by a built-in mechanism with fuzzy logic connectives. To facilitate rule editing, fuzzy concepts can be modified on-screen by changing critical points in the domain over which the associated membership functions are defined.

In the display module, images can be displayed directly by associating colors with spectral bands without classification, in order to support direct visualization of the preclassified scene. However the most important component of the module supports the display of classified images. In general, techniques of dithering and bit-mapping can be used to display uncertainty (Leung and Leung, 1990) in terms of levels of class membership, to expose the spatial variation in membership within regions or across region boundaries. In addition the system provides several other measures and methods for conveying information about an MPF to the user. The following sections briefly describe the principal tools.

1. Unclassified image

Colors can be assigned to spectral bands to create conventional false-color representations of the unclassified scene. This allows the user to see the raw data before classification.

2. Classified image

The RGB color model is used to display the results generated by the fuzzy classifier, or input from some other source. Each class is associated with a point in RGB space, and each vector of class memberships is mapped to an intermediate point in the color space by linear interpolation. This method is successful for two classes ($n=2$) provided the pure-class colors are chosen carefully, but it is difficult for the eye to decode the results for $n=3$, and for $n>3$ the mapping from class membership vector to color space is no longer unique. Moreover mapping is non-unique for $n=3$ if the class

memberships have not been normalized to sum to 1 (see above).

It is possible to display each pixel's degree of belonging to each class as a numerical value, or graphically as a bar chart. The corresponding location in color space can also be displayed.

To deal with the difficulty of visualizing membership in many classes, it is possible to display each class's memberships separately using a grey scale. By using multiple windows one can display the general distribution of each class for up to four or even six classes simultaneously.

Sometimes it is desirable to have a non-fuzzy image of a fuzzy scene. A simple defuzzing mechanism is maximum likelihood, where the displayed class $f(x) = i$ if $\pi_i(x) > \pi_j(x)$ for all i, j , i not equal to j ; that is, a pixel is assigned to class i (and displayed with class i 's color) if its degree of membership in class i is highest. The user has control over the colors assigned to each class. Frequency distributions of the entire image can be displayed, and the user can zoom into a selected area, or display the contents of any pixel.

3. Area

Calculation of the area occupied by each class is a common GIS function. For conventionally classified scenes or other forms of raster data it is calculated by counting the pixels assigned to each class and multiplying by pixel area. However the solution is less clear in the case of fuzzy-classified scenes. If $p_i(x)$ is interpreted as the proportion of pixel x that is truly class i , as in a mixed pixel interpretation of fuzziness, then the area of class i will be the sum of such fractions added over the scene. On the other hand if $p_i(x)$ is interpreted probabilistically, the same estimate must be interpreted as the expected area of class i . Similar approaches are appropriate if $p_i(x)$ is given other probabilistic interpretations. Thus the calculation of area on a fuzzy-classified scene seems adequately addressed by calculating:

$$A_i = b \sum_x p_i(x)$$

where b is the area of each raster cell.

More difficult is the estimation of error variance, standard error, or the uncertainty associated with such estimates. In the mixed pixel interpretation A_i is deterministic, with zero uncertainty. In a probabilistic interpretation, and assuming that outcomes in each

pixel are independent of outcomes in neighboring pixels (zero spatial dependence) then the uncertainty associated with area estimates can be determined from the statistics of the binomial distribution in the form of a standard error:

$$s_{ei} = b \left\{ \sum_i p_i(x) [1-p_i(x)] \right\}^{1/2}$$

where s_{ei} is the root mean square uncertainty in estimate A_i (Fisher, 1991, used Monte Carlo simulation to estimate standard error). But when spatial dependence is present, as it almost always is, and outcomes in neighboring pixels are correlated, it is necessary to resort to the methods described by Goodchild, Sun and Yang (1992).

4. Entropy

The degree of certainty in a pixel's classification can be measured in various ways, but one that expresses the degree to which membership is concentrated in a particular class, rather than spread over a number of classes, is the information statistic or entropy measure:

$$H(x) = - (1/\ln n) \sum_i p_i(x) \ln p_i(x)$$

where $H(x)$ is the entropy associated with pixel x . $H(x)$ varies from 0 (one class has probability 1, all others have probability 0) to 1 (all classes have probability equal to $1/n$). The system allows a map of H to be displayed using a grey scale; light areas have high certainty (probability concentrated in one class) while dark areas have low certainty.

The degree of fuzziness associated with membership in each class can be assessed by another form of the entropy measure:

$$H_i = - (1/N \ln 2) \sum_x \{ p_i(x) \ln p_i(x) + [1-p_i(x)] \ln [1-p_i(x)] \}$$

where the sum is now over the pixels and N is the number of pixels. H_i is zero if the probability of membership in class i is 0 or 1 in all pixels, and 1 if probability is 0.5 in all pixels. The overall entropy H of the entire fuzzy scene can be obtained by adding these measures over all classes.

5. Realizations

As noted earlier, an important aspect of visualizing uncertainty is the ability to view individual realizations

of an error model, rather than its parameters. All of the previously noted methods display some aspect of the probability vectors, which are the parameters of the error model's stochastic process, rather than its outcomes. Viewing a display of probability vectors necessarily diverts attention from the variation between realizations, and focuses more on the average or expected case.

The system includes the ability to display realizations of the error model, using user-determined levels of spatial dependence. Goodchild, Sun and Yang (1992) discuss possible methods for determining appropriate levels, as attributes of the entire map, or of individual classes, or of geographic regions. A display of four or six different realizations in different windows on the screen provides graphic illustration of the implications of uncertainty in spatial data, and draws attention to its influence on analysis, modeling and decision-making.

Summary and Future Directions

It is often argued in the GIS community that while uncertainty is endemic to spatial data and undoubtedly affects the outcomes of spatial data processing, it is best not to draw attention to it because of its complexity and potentially damaging effects on decision-making. The user "does not want to know". Analogous software systems, such as the statistical packages and database management systems, do not include techniques for capturing, storing and manipulating explicit information on uncertainty, so why should GIS? We believe that this argument is both intellectually unsound and disastrously shortsighted. Most spatial decisions, particularly important ones, are made in an environment of conflict and controversy. As GIS matures and becomes available to more and more parties to a debate, the naive view that the party with the GIS somehow carries greater weight will become less and less realistic, and easier and easier to attack. Pressures for better quality assurance and control are already emerging from instances of GIS-related litigation.

Spatial statistics is a complex and difficult field, and few GIS practitioners have more than an elementary understanding of its techniques and concepts. Moreover visual techniques are inherently convincing and communicative. Thus it seems that visualization will have to be a fundamental part of any concerted effort to handle uncertainty within GIS. Goodchild, Sun and Yang (1992) have argued that visualization is the key to user participation in the determination of the key spatial dependence parameters in spatial statistical models of uncertainty.

An MPF is inherently multidimensional, and this paper has presented a number of techniques for improving the user's ability to understand this particular form of spatial variation. However any communication system must satisfy the requirements of the user as much as it exploits the capabilities of the system, and it seems clear to us that an ideal design can only come from the experience of working with these tools in a real analytic environment.

Acknowledgment

The National Center for Geographic Information and Analysis is supported by the National Science Foundation, grant SES-10917.

References

- Bezdek, J.C., 1981. *Pattern Recognition with Fuzzy Objective Function Algorithms* New York: Plenum.
- Borden, F.Y., Applegate, D.N., Turner, B.J., Merembeck, B.F., Crenshaw, E.G., Lachowski, H.M. and Thompson, D.N., 1977. "Satellite and Aircraft Multispectral Scanner Digital Data Users Manual" *Technical Report ORSER-SSEL 1-77*, University Park, PA: Pennsylvania State University.
- Coleman, G.R. and Andrews, H.C., 1979. "Image Segmentation by Clustering" *Proceedings, IEEE* 67: 773-785.
- Duda, R.O. and Hart, P.E., 1973. *Pattern Classification and Scene Analysis*, New York: Wiley.
- Englund, E., 1992. "Spatial Simulation: Environmental Applications" in Goodchild, M.F., Parks, B.O. and Steyaert, L. (eds.) *GIS with Environmental Modeling*, New York: Oxford University Press (forthcoming).
- Fisher, P.F., 1991. "Modeling Soil Map-Unit Inclusions by Monte Carlo Simulation" *International Journal of Geographical Information Systems* 5(2): 193-208.
- Goodchild, M.F., Sun, G. and Yang, S., 1992. "Development and Test of an Error Model for Categorical Data" *International Journal of Geographical Information Systems* (forthcoming).

- Gopal, S. and Woodcock, C., 1992. *Accuracy Assessment of Thematic Maps Using Fuzzy Sets 1: Theory and Methods* (unpublished paper).
- Laurini, R. and Thompson, D., 1992. *Fundamentals of Spatial Information Systems*, San Diego, CA: Academic Press, 680pp.
- Leung, Y., 1984. "Towards a Flexible Framework for Regionalization" *Environment and Planning A* 16: 203-215.
- Leung, Y., 1985. "A Linguistically-Based Regional Classification System" in Nijkamp, P., Leitner, H. and Wrigley, N. (eds.) *Measuring the Unmeasurable*, Dordrecht: Martinus Nijhoff, pp. 451-486.
- Leung, Y., 1987. "On the Imprecision of Boundaries" *Geographical Analysis* 19: 125-151.
- Leung, Y. and Leung, K.S. (1990) *Analysis and Display of Imprecision in Raster-Based Information Systems* (unpublished paper).
- Leung, Y. and Leung, K.S. (1992a) An intelligent expert system shell for knowledge-based geographic information systems: 1, the tools. *International Journal of Geographical Information Systems* (forthcoming).
- Leung, Y. and Leung, K.S. (1992b) An intelligent expert system shell for knowledge-based geographic information systems: 2, some applications. *International Journal of Geographical Information Systems* (forthcoming).
- Mark, D.M. and Csillag, F., 1989. "The Nature of Boundaries in 'Area-Class' Maps" *Cartographica* 26(1): 65-78.
- Nilsson, N.J., 1965. *Learning Machines: Foundations of Trainable Pattern-Classifying Systems*, New York: McGraw Hill.
- Phillips, T.L. (ed.), 1973. *Larsys Version 3 Users Manual* West Lafayette, IN: Laboratory for Applications of Remote Sensing, Purdue University.
- Ruspini, E.H., 1970. "Numerical Methods for Fuzzy Clustering" *Information Sciences* 319-350.
- Ruspini, E.H., 1973. "New Experimental Results in Fuzzy Clustering" *Information Sciences* 273-284.
- Tomlin, C.D., 1991. *Geographic Information Systems and Cartographic Modeling*, Englewood Cliffs, NJ: Prentice-Hall.
- Wacker, A.G. and Landgrebe, D.A., 1972. "Minimum Distance Classification in Remote Sensing" *First Canadian Symposium on Remote Sensing, Ottawa*.

Integrating Maps and Statistical Graphs in Graphic Scripts*

Mark Monmonier

Department of Geography
Syracuse University
Syracuse, NY 13244-1160
mon2ier@sunrise.acs.syr.edu

Abstract

Graphic scripts are automated sequences of maps, text blocks, statistical diagrams, and graphs. Graphic scripts can be orchestrated to introduce a data set, explore salient trends, or reveal anomalies. A graphic script can include graphic phrases, which are programmed, data-driven graphic sequences with a specific focus, such as presenting and exploring the geographic pattern of residuals for a bivariate regression. Graphic scripts and phrases can have important roles in exploratory analysis as training aids and programmed overviews. As expository tools in an exploratory environment, graphic scripts and phrases require a comparatively smooth, logically structured flow of informative views analogous to the flow required for clarity in expository prose. Experience in developing two graphic scripts, one addressing bivariate correlation and the other exploring spatial-temporal trends, suggests the existence of four basic strategies for using sequence and motion to integrate maps and statistical graphs. The 'graph-then-map' strategy follows a graph addressing the overall region (such as a frequency graph for the United States as a whole) with a map addressing the constituent subregional units (such as a map of the states). The 'graph-as-a-key' strategy links a map to a graph that serves as the map key and uses a programmed examination of this key to generate an animated exploration of spatial trends. In contrast, the 'map-as-a-key' strategy uses an animated region-by-region examination to explore spatial patterns or statistical relationships. The 'complementary-juxtaposition' strategy places two (or more) major graphics on the screen for simultaneous viewing. Neither graphic is markedly subordinate, and if the display is dynamic, its constituent graphics are linked and synchronized. The paper also discusses signature hues and the roles of text and speech.

Introduction

Dynamic graphics can be user-driven or system-driven. Most contemporary systems for interactive statistical graphics are user-driven: except for the simple spin rotation of the point cloud in a three-dimensional scatterplot, the analyst uses a menu and pointing device to select every view, or change of screen. In contrast, a system-driven dynamic presentation could consist of a logical or traditional sequence of graphics, such as juxtaposed frequency diagrams for a pair of variables followed by a scatterplot to which the system adds first a least-squares regression line and then a smooth curve generated by a lowess operator (Cleveland 1979). In its broadest sense, a system-driven dynamic sequence can be responsive to the goals of the analyst, the method of data collection, and trends and anomalies in the data. The user might control the pace of the presentation, and even alter its direction to recall a previous screen. A system-driven sequence of statistical graphics can range in complexity from a simple rotating point cloud to the computationally sophisticated technique called 'the grand tour' (Buja and Asimov 1986) to an even more varied script designed to present a thorough visual analysis of a large and complex data set.

Scripted sequences of views can be particularly useful in exploring geographic data. These sequences can vary widely in sophistication and duration. At a basic level, a highly focussed *graphic phrase* can describe a single variable by generating a complementary set of maps that avoids the analytic myopia and graphic inadequacy of the traditional one-map cartographic solution (Monmonier 1989). Another kind of graphic phrase might link a cartographic view of the data to a statistical view, such as a frequency histogram or scatterplot. At a higher level, a more comprehensive *graphic script* might generate a relational sequence of maps and statistical diagrams that first explores the individual variances of two variables, then their cross-correlation, and finally the residuals resulting from fitting one distribution to

* The author gratefully acknowledges the support of a National Science Foundation grant (SES-90-22845).

the other. Graphic scripts might include graphic phrases that examine the effects of areal aggregation and spatial autocorrelation, and a *data profile* could direct the sequence of graphics to address uncertainty in the data, problems of measurement and definition, and relevant complementary relationships with related variables. Furthermore, a *user profile* could ensure a graphic sequence attentive to the user's experience, interest in specific places, and research objective, however vaguely defined.

The goal of these tools for geographic visualization is thoroughness and efficiency, not the replacement of serendipitous discovery with a hurried, banal sequence of dazzling, Nintendo-like images. I call this approach 'atlas touring' because it can orchestrate a tailored, user-relevant introduction or overview of a geographic database, sometimes called an 'electronic atlas'. Although atlas touring might provide some users with a graphic summary (Monmonier 1992b), it could serve others as an intellectual stimulus, electronic tour-guide, or analytical pump-primer.

This paper is an early attempt to develop a theory for authoring graphic scripts. It draws on two recently developed prototype scripts, one designed to examine relationships among two geographic distributions, and the other intended to examine spatial-temporal trends for a single historical series of measurements. After a brief description of these two graphic scripts and some of their constituent graphic phrases, I discuss four basic strategies for using sequencing and motion to integrate the geographic-space representation of the map with the attribute-space representation of the statistical diagram.

Two Prototype Graphic Scripts

Because no one had devised a graphic sequence with the functionality and visual variety I envisioned, I chose to begin by addressing two relatively straightforward geographic problems for which I had data. The sequence I call the 'correlation script' looks at two variables, females as a percentage of elected local public officials and the proportion of civilian women 16 and older in the labor force. Both variables were measured for 1987 for the 50 states of the U.S. I assigned the first variable the brief title "Female Officials" and treated it as the dependent variable. And the second variable I called "Females Working" and employed as the independent variable. My second prototype, which I call the 'historical script', examines temporal change for a single variable, the number of daily newspaper firms recorded annually

for each of the 91 years from 1900 through 1990. For simplicity I merged counts for the District of Columbia with those for Maryland, and used territorial data for Arizona, New Mexico, and Oklahoma, which were not yet states at the beginning of the period.

I organized each script into acts and scenes. According to Philip Gersmehl (1990), a geographer who identified nine animation metaphors, this 'stage and play' strategy is especially appropriate for complex scripts with a geographic setting. The first act introduces each variable in a separate scene. Because the prototype scripts have no sound track, each introductory scene begins with a large block of text that lists the variable by both its full and abridged titles, defines the variable concisely, and mentions important exceptions or refinements. Like the correlation script, the historical script employs two primary variables, a raw count described by the brief title "Number of Firms" and a derivative variable with the abbreviated name "Pct. of Maximum." This second measure represents each yearly count as a percentage of the maximum annual count recorded for the state during the 91 years of record.

I extended the stage-and-play metaphor by dressing each variable in a signature hue used to color area-fill patterns on its maps, bars on its histograms, and brushes on its scatterplots. Contrast is important in selecting signature hues. For the correlation script I used red for "Female Officials," blue for "Females Working," and magenta for the residuals from regression introduced in the third act. In the historical script magenta serves as the signature hue for

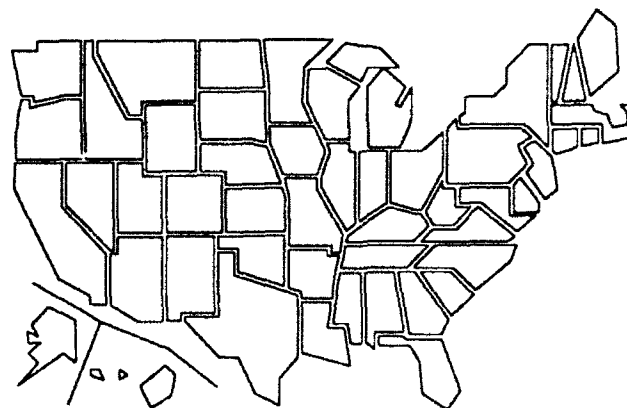


Figure 1. Visibility base map represents states with noncontiguous, computationally simple geographic caricatures.

"Number of Firms" and cyan represents "Pct. of Maximum" because I reserved red and blue to differentiate gains from losses in an extensive series of maps portraying both absolute change and rates of change.

Like both the cost-conscious producer and the director wary of confusing the audience, I avoided complexity by using concise labels and titles; by establishing standard positions for maps, diagrams, keys, and blocks of text; and by employing a visibility base map (Figure 1) with caricatured state polygons enlarged to provide readable area-fill patterns for Delaware, Rhode Island, and other small states. (Graphic parsimony has other advantages: by keeping the number of props to a minimum and assigning them fixed places on the screen, I also conserved memory and simplified programming.)

Sequencing is important at two levels. At the macro level, the plots of both graphic scripts move progressively yet carefully toward a climax. The correlation script, which examines each variable separately in the first act and both variables jointly in the second act, culminates in the third and final act with a revealing dynamic examination of the geographic pattern of residuals from regression. Similarly, the historical script, which examines its count and percentage indicators separately in the first act and addresses spatial-temporal patterns of change in the second act, caps the presentation with a sequence of dynamic centographic maps that relate change in the newspaper industry to the westward and southward advance of nation's center of population. At the micro level, each graphic phrase requires a coherent, carefully paced succession of images, in which new information is added to older information one piece at a time. For example, before initiating a cartographic animation the screen might reveal in discrete steps separated by noticeable pauses (1) the base map to be used, (2) a text block describing the nature or point of the graphic phrase, (3) a map title to identify the variable and its treatment, (4) the static elements of the map key, and (5) any temporally variable elements of the key or title, such as the year-date or a linear time scale.

Text is an important element in the two prototype graphic scripts. As narratives presented without a programmed audio channel, the scripts use text screens and small part-screen text blocks in the same way that silent films employed *intertitles* to establish a context for the action to follow and to supply background information not easily woven into the plot by other means (Fleishman 1992, 23-24). Thus, text

screens such as Figure 2 use natural language to the describe the statistical-cartographic narrative's lead characters, map titles and axis labels to identify these characters in their various guises, and small text blocks at the top or bottom of the screen to announce the intent of the scene or graphic phrase to follow.

Adding a sound channel would promote a fuller, more efficient integration of narrative text and narrative action. Because the viewer cannot simultaneously read text and watch animated graphics, my next stage of development will use recorded or synthesized speech to supplement text blocks and labels in some cases and to replace them in others. But even where listening might replace reading, careful sequencing will be important because the viewer often requires a verbal explanation to comprehend the action that has begun or is to follow. Indeed, graphic scripts can use a meticulously synchronized audio channel in at least three ways: (1) to announce what the viewer will see shortly, (2) to describe or interpret what she is seeing currently (or should be able to see,) and (3) to summarize what she has just seen.

Strategies for Integrating Maps and Graphs

The remainder of this paper examines four strategies for linking maps and statistical graphs. Devised while preparing the two prototype graphic scripts, these approaches are useful for fully authored scripts as well as for graphic phrases provided as dynamic analytical tools in an interactive system for exploratory data analysis. Although this four-strategy framework does not exhaust the range of integrating mechanisms, it offers script authors and software designers a broad variety of choices. The discussion that follows also examines useful variations within each strategy.

1. The Graph-then-Map Strategy

The simplest and most obvious approach to map-graph integration is the graph-then-map sequence in which a graph presenting an overview or summary for the entire study region precedes a map showing information for the region's component parts. This approach parallels most written and oral narratives by examining geographic details immediately after providing a more general context for the map's reading and interpretation. Thus, for a spatial-temporal distribution a time-series graph might usefully point out whether the snapshot map that

follows reflects a period of relative growth, decline, or stability. For a map portraying the geographic pattern of change over a stated period, a preliminary time-series graph might either (1) integrate a relatively short span of time represented by the map into a longer historical record or (2) describe the coherence, volatility, or cyclic character of the period at a finer level of temporal resolution. Among other examples, graphs preceding maps might inform the viewer about disparities among subregions or about differences apparent when demographic or socioeconomic data are disaggregated by gender, race, ethnicity, income, or place of residence (urban or rural).

Early scenes in the historical script illustrate the importance of the graph-then-map strategy in dynamic cartography. Introductory scenes in the first act introduce the script's two main variables with an animated time-series graph for the entire United States (Figure 3) in which vertical bars are added rapidly, year by year, from left to right. A short blink-sequence then highlights the minimum and maximum values and their positions within the 91-year period. The scene then uses a rapid series of cartographic snapshots for individual years to portray the historical evolution of the variable's geographic pattern. As Figure 4 illustrates, a sufficiently large monitor allowed the addition of this dynamic map to the same screen, above the time-series graph. An identical signature hue, which colors both the bars of the graph and the interior variable-height indicator bars of the map's frame-rectangle symbols, also promotes integration of the graph and map.

2. The Graph-as-a-Key Strategy

As its name implies, the graph-as-a-key strategy employs a statistical graph as the map's key, or legend. As with most map keys, the graph usually would be subordinate to the map in role, complexity, position, and size. Figure 5, used in the introductory scenes of the correlation script, is a good example of this relationship. The vertical bar graph below the map describes the numerical distribution of data values, arranged in rank-order from left to right. Each state accounts for one polygon on the map and one bar in the graph. Bars and polygons are linked so that when a quintile (equal-fifths) classification is imposed on the map, the graph serves as the map key. More informative than the standard key of a choropleth map, the bar graph illustrates both the internal homogeneity and size (number of members) of each of the map's five categories.

The graphic phrase in which Figure 5 appears also uses dynamic blinking to reinforce the link between map and graph. A canvass-by-category blink sequence highlights each of the map's five categories, starting with the highest category represented by the rightmost group of bars and their linked polygons, and moving in sequence to the lowest category. Blinking consists of simultaneously changing the interiors of all bars and polygons for the category from the unaltered fill pattern in Figure 5 to white and back again. (For contrast, the white fill for the lowest category alternates with solid black.) Blinking is rapid: despite ten simultaneous alternations for each category, the entire five-category canvass takes no more than 15 seconds.

The first two scenes of the correlation script introduce the link between map and graph with a simpler, more elementary two-category map. Immediately following a text screen similar to Figure 2, the system juxtaposes a blank map and graph otherwise similar to those in Figure 5. Action begins with the upward-sweep-by-rank sequence described by the screen snapshot in Figure 6. The single category break moves swiftly across the graph from left to right at a uniform rate, filling all the bars in its wake with the variable's signature hue—red for the dependent variable or blue for the independent variable. Because each polygon is linked to a bar in the graph, the instantaneous view in Figure 6 shows the states with the 13 lowest values. After this upward sweep by rank fills all bars and polygons with the variable's signature hue, a downward sweep reverses the process by moving the break back across the graph to the left. The next graphic phrase is an upward sweep *by value*, in which the level of the thermometer-like indicator to the left of the bar graph rises at a uniform rate so that periods of relative inaction in both map and graph reflect gaps in the distribution. Again, after the sweep reaches the rightmost bar and the top of the indicator, the system sweeps backward toward the minimum data value at the lower left.

A variation of the graph-as-a-key strategy takes advantage of the time-series graph in Figures 3 and 4. In the historical script's second act, several dynamic maps address the direction and amount of change for various time periods within the 91 years of record. The first of these cartographic animations treats only the occurrence and direction of change. As Figure 7 illustrates in monochrome, the map's simple, three-pattern key describes change with the labels "gain", "loss", and "same", which the prototype script portrays in red, blue, and white, respectively. As the

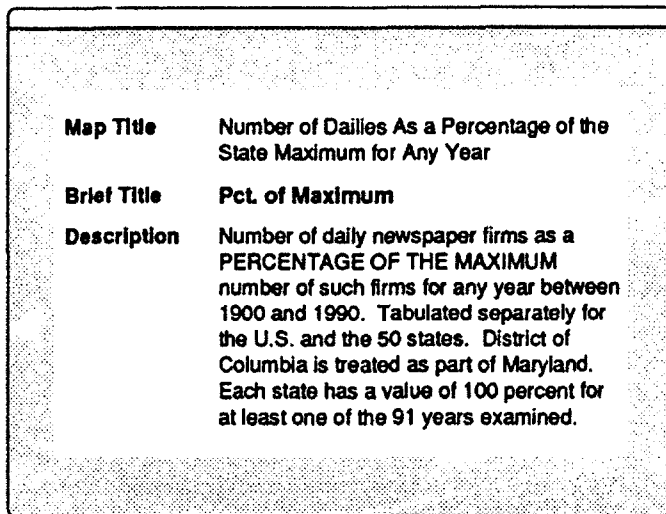


Figure 2. Text screen names and describes the second principal variable in the historical script. (Graphics for this paper were produced with MacDraw II to provide more aesthetic and readable versions of color screens generated in real time on a Macintosh II by programs written in Think C.)

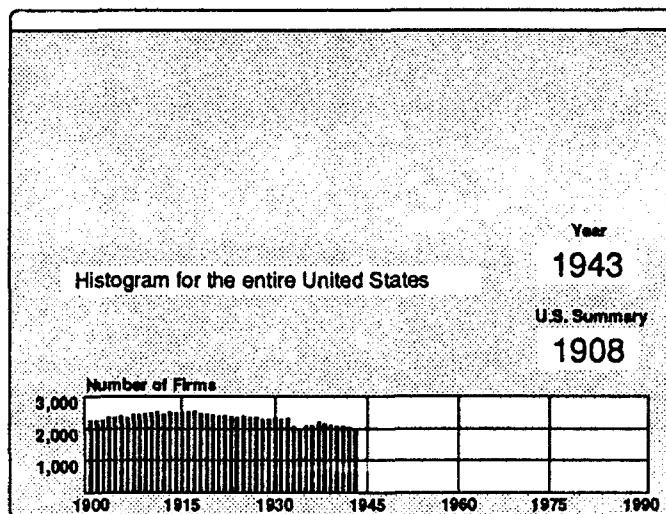


Figure 3. This dynamic time-series graph precedes a dynamic spatial-temporal map in the introductory scenes of the historical script. Screen snapshot freezes the action for 1943 in a rapid dynamic sequence reaching from 1900 to 1990.

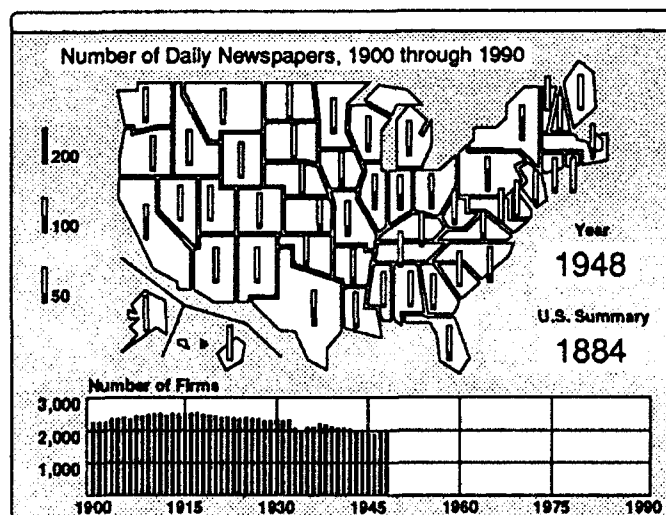


Figure 4. This dynamic spatial-temporal map shares the screen with the dynamic time-series graph that precedes it. As the dynamic sequence moves forward in time, the system regenerates the graph below the map.

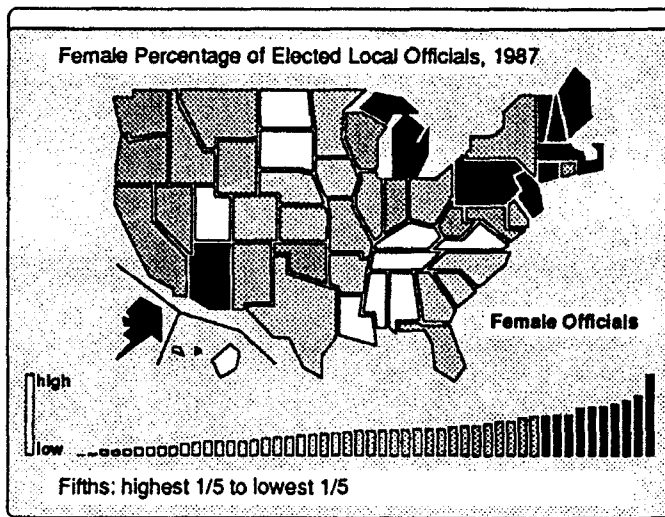


Figure 5. Bar graph serves as key for a five-category choropleth map in the correlation script's first act. In the color version, the bars and polygons for the second through fifth categories appear in red, the signature hue for the dependent variable.

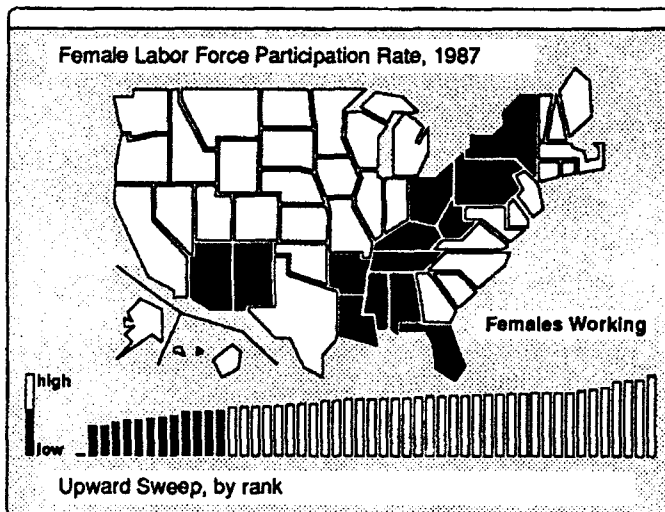


Figure 6. Snapshot of an upward sweep of the bar graph shows the geographic pattern of the 13 lowest states for the correlation script's independent variable. On a color monitor, the signature hue blue fills the darkened bars and polygons.

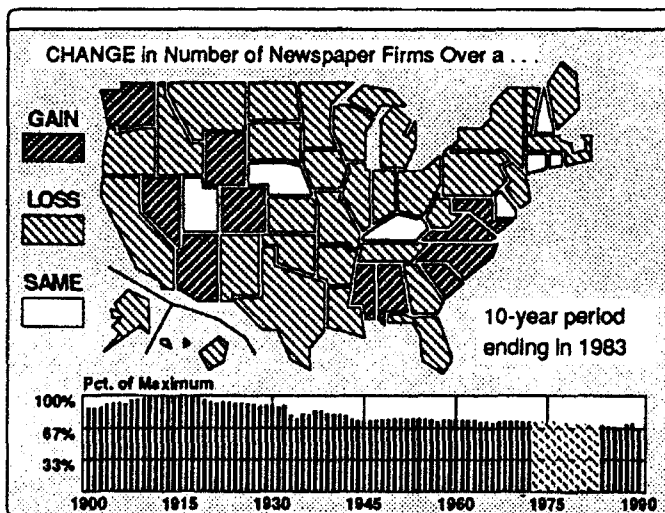


Figure 7. Contiguous group of colored bars moves across the time-series graph below the map to describe not only the position and duration of the time period portrayed on the map but also the overall, nationwide change for the period. In this monochrome version of a screen displayed in color, the darker, upward-slanting diagonal patterns represent red and the lighter, downward-slanting patterns represent blue. The full-color image uses solid hues (red, blue, white) for the three categories.

time period represented by the map moves forward from 1900 to 1990, a color reflecting the corresponding national pattern for the instantaneous time period advances across the graph from left to right. On the screen represented in Figure 7, the contiguous group of bars for 1973 through 1983 would be blue because of the nationwide decrease during this ten-year period. But for an equivalent view near the beginning of the century, the colored group of bars would turn red to show, for example, an overall increase from 1902 to 1912. In this way the graph serves as a second key to describe not only the length and position in time of the period portrayed on the map but also a net gain, loss, or sameness for the nation as a whole.

3. The Map-as-a-Key Strategy

Sometimes the graph is more central to a graphic sequence than its accompanying map. The historical script employs this map-as-a-key strategy in a canvass-by-region sequence that examines the temporal trends of its two principal variables. In this graphic phrase a separate dynamic sequence addresses each of the nine regional divisions recognized by the Bureau of the Census. As Figure 8 illustrates for the South Atlantic Division, the map at the top of the screen highlights the states comprising the region, while the two time-series graphs immediately below the map describe temporal change in the region, and a time-series graph at the bottom of the screen puts the lower of the two regional time-series in perspective by presenting the same variable for the nation as a whole. The map is constant for each region, but the graphs are animated through the simultaneous addition of vertical bars from left to right as the year-date window advances rapidly in time, year by year, from 1900 to 1990. The bars are color coded with each variable's respective signature hue: magenta for the upper graph and cyan for the middle and bottom graphs. A region-name window to the right of the map supplements the pattern of highlighted polygons.

Figure 9 demonstrates a similar use of the map as a key for the graph. In the final act of the correlation script, a graphic phrase presents a scatterplot that includes a data point for each state and a least-squares regression line summarizing the modest ($r = 0.31$) general relationship between the two variables. A canvass-by-region sequence then locates each regional division in geographic space by highlighting its states on the map while the scatterplot locates the same states in bivariate attribute space by temporarily suppressing the data-point symbols for all other states.

In the snapshot in Figure 9, a clustering of points above the regression line indicates that the five states of the East North Central division have a somewhat greater tendency to elect females to local public office than their proportions of females in the labor force would suggest.

Although graphic phrases canvassing the study area region by region will account for most applications of the map-as-a-key strategy, the map could also portray one of a series of trend directions or control variables described on a richer, more interesting accompanying graph.

4. The Complementary-Juxtaposition Strategy

Juxtaposed maps and graphs can be so closely integrated that neither dominates the other during an act or scene. Both elements might change simultaneously, or the focus of animation might alternate several times throughout the scene. The second act of the correlation script provides a good example of complementary juxtaposition with a screen layout that includes a scatterplot and separate maps for each variable. The initial scene uses automated brushing to explore first the dependent variable and then the independent variable. Figure 10 illustrates the horizontal brush (Becker and Cleveland 1987) that moves up and down the scatterplot along the vertical axis, which represents the dependent variable. The upper map, which also represents the dependent variable, highlights all polygons linked to points instantaneously within the brush. After two cycles of an upward stroke followed by a downward stroke, the brush rotates 90 degrees to explore the horizontal axis, representing the independent variable. During this sequence of horizontal strokes, the lower map highlights states instantaneously selected by the vertical scatterplot brush. Throughout this scene signature hues promote map-graph integration as the upper map highlights polygons in red when a screened-reddish brush travels along the vertical axis and the lower map highlights polygons in blue when a screened-bluish brush explores the horizontal axis.

Other scenes in the correlation script use simultaneous motion to link maps and graphs. Later in the second act, for instance, a four-hue bivariate cross-classification map (Dunn 1989) shares the screen with a four-hue scatterplot partitioned into four regions, each linked to one of the map's four categories. Although the scatterplot serves as the map's key, a graphic sequence that explores the stability of the variables' geographic covariance by

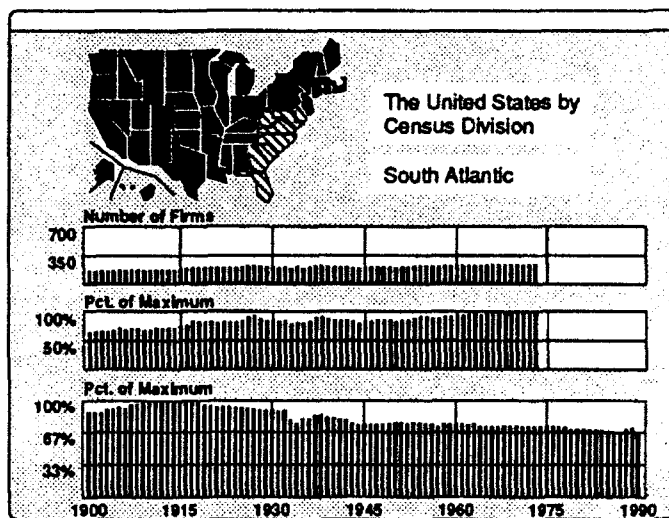


Figure 8. Map identifies the region represented by the middle two dynamic time-series graphs. Signature hues color the histogram bars: solid magenta for the upper histogram and solid cyan for the two lower histograms.

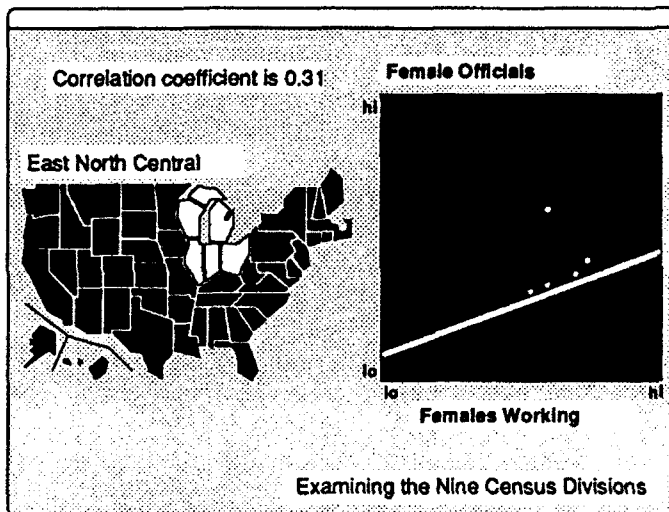


Figure 9. Map points out the states comprising the region highlighted on the accompanying scatterplot. Non-highlighted polygons and scatterplot background are in black, highlighted polygons and scatterplot dots are in yellow, and the regression line is in magenta.

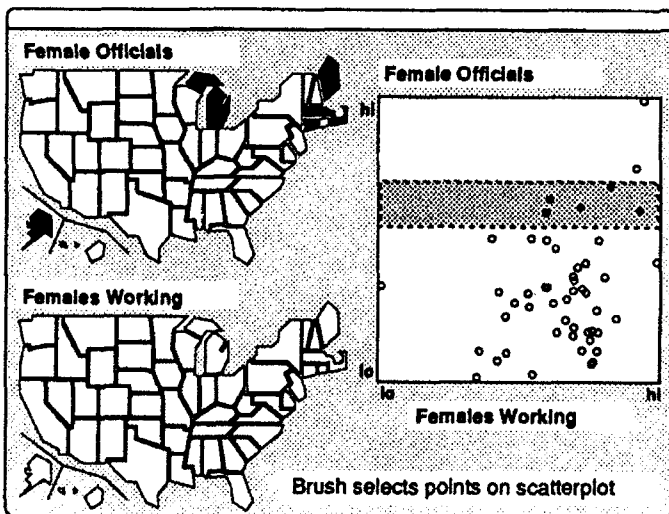


Figure 10. As a horizontal brush explores the scatterplot's vertical axis, the upper map highlights polygons linked to points instantaneously within the brush.

varying the category breaks produces simultaneous motion on both map and scatterplot. And in the correlation script's final act, as another example, a joint, category-by-category canvass of a scatterplot and a map of residuals dynamically links complementary, juxtaposed views of bivariate-attribute space and geographic space.

Simultaneous motion on juxtaposed maps and graphs might easily tax the viewer's ability to process visual information. Yet closer examination of the examples discussed here reveals that predictable, script-driven action in one element allows the viewer to concentrate on less systematic, data-driven action in the other element. In Figure 10, for example, 'automated brushing' of the scatterplot lets the viewer focus on the map while merely keeping track of the direction and relative position of the scatterplot brush. In the two examples not illustrated, the map is also the focal element. But were a geographic brush to sweep across the map from north to south, perhaps in an application concerned with climatology or skin cancer, the viewer could focus on the scatterplot, instead of the map. The need for a single, coherent focus might explain why the historical script, which is based on an inherently more complex set of spatial-temporal data, provides no examples of the complementary-juxtaposition strategy.

Concluding Remarks

I want to conclude with two additional points. First, although the foregoing discussion of map-graph integration might imply that visual objects to be integrated are either maps or aspatial statistical graphs, a graphic script might well include a hybrid graphic that assigns one of the graphic plane's two dimensions to an attribute and the other to distance. I have described the use of such 'half maps' elsewhere, in the context of static representations of geographic data (Monmonier 1988) and spatial-temporal data (Monmonier 1990). In a dynamic context, either sequencing in time or juxtaposition in space can position a hybrid map-graph between a full map and a full graph. Indeed, any of the four strategies outlined here can further integrate such a hybrid into a graphic script and thereby enhance its role as a visual bridge between statistical-graphic and cartographic representations.

My second point is that coherent and effective integration of maps and graphs in a graphic script requires careful planning and editing. For the same reasons that journalists and nonfiction writers must

avoid run-on sentences and choppy prose, authors of graphic scripts must consciously avoid overly busy screens and erratically unpredictable graphic sequences. Indeed, graphic scripts can benefit from the same exact stylistic principles that promote lucid narrative prose (Monmonier 1992a). These principles include the early announcement of the narrative's point, or purpose, and a smooth, coherent flow from old information to new information (Williams 1990). Signature hues, consistent labels, and text windows describing a graphic phrase's goal or purpose also contribute to this coherence. Indeed, the visually rich and complex sensory environment of a multimedia cartographic presentation requires logical and consistent sequences, symbols, and screen layouts, which the viewer can rapidly decode and comprehend. Synthesized or digitally recorded speech might make these devices even more effective by making their roles more apparent to the user. After all, as a narrative approach to information retrieval and analysis, the graphic script can benefit greatly from a fuller integration of graphics and natural language.

References

- Becker, Richard A., and Cleveland, William S. (1987) "Brushing Scatterplots," *Technometrics*, 29: 127-142.
- Buja, Andreas, and Daniel Asimov (1986) Grand tour methods: an outline. *Computer Science and Statistics: The Interface*, ed. D. M. Allen, pp. 63-67. Amsterdam and New York: Elsevier Science Publishers.
- Cleveland, William S. (1979) Robust locally weighted regression and smoothing scatterplots. *Journal of the American Statistical Association*, 74: 829-836.
- Dunn, Richard (1989) A dynamic approach to two-variable color mapping. *The American Statistician*, 43: 245-252.
- Fleishman, Avrom (1992) *Narrated Films: Storytelling Situations in Cinema History*. Baltimore: Johns Hopkins University Press.
- Gersmehl, Philip J. (1990) Choosing tools: Nine metaphors of four-dimensional cartography. *Cartographic Perspectives*, no. 5, pp. 3-16.
- Monmonier, Mark (1988) Geographical representations in statistical graphics: a conceptual framework.

American Statistical Association, Proceedings of the Section on Statistical Graphics, 1988, pp. 1-10.

Monmonier, Mark (1989) Graphic scripts for the sequenced visualization of geographic data. *Proceedings of GIS/LIS '89, Orlando, Florida, Nov. 26-30, 1989*, pp. 381-389.

Monmonier, Mark (1990) Strategies for the visualization of geographic time-series data. *Cartographica*, 27, no. 1: 30-45.

Monmonier, Mark (1992a) Authoring graphic scripts: Experiences and principles. *Cartography and Geographic Information Systems*, 19: forthcoming

Monmonier, Mark (1992b) Summary graphics for integrated visualization in dynamic cartography. *Cartography and Geographic Information Systems*, 19: 23-36.

Williams, Joseph M. (1990) *Style: Toward Clarity and Grace*. Chicago: University of Chicago Press.

Iterative and Non-iterative Simulation Algorithms

Andrew Gelman*
Department of Statistics
University of California
Berkeley, CA 94720

Abstract

The Gibbs sampler, Metropolis' algorithm, and similar iterative simulation methods are related to rejection sampling and importance sampling, two methods which have been traditionally thought of as non-iterative. We explore connections between importance sampling, iterative simulation, and importance-weighted resampling (SIR), and present new algorithms that combine aspects of importance sampling, Metropolis' algorithm, and the Gibbs sampler.

1. Introduction

Currently, one of the most active topics in statistical computation is inference from iterative simulation, especially the Metropolis algorithm and the Gibbs sampler (Metropolis and Ulam, 1949; Metropolis et al., 1953; Hastings, 1970; Geman and Geman, 1984; Gelfand et al., 1990). (The Gibbs sampler is in fact a special case of the generalized Metropolis algorithm; see Section 4.3 below.) The essential idea of iterative simulation is to draw values of a random variable x from a sequence of distributions that converge, as iterations continue, to the desired *target distribution* of x . For inference about x , iterative simulation is typically less efficient than direct simulation, which is simply drawing from the target distribution, but iterative simulation is applicable in a much wider range of cases, as current statistical literature makes abundantly clear (see, e.g., Smith and Roberts, 1993, Besag and Green, 1993, and Gilks et al., 1993).

This paper presents iterative simulation methods as an outgrowth of the non-iterative methods of rejection and importance sampling, both of which use simulation to correct an approximation of the target distribution.¹ The connection between iterative and non-iterative simulation is of interest for three reasons. First, a unified treatment is appealing, and highlights the similar prob-

lems faced by users of all of these methods. Second, the general formulation suggests potentially useful new methods, such as the iterative importance resampling of Section 3.3 and the Metropolis-approximate Gibbs sampler of Section 4.4. Third, noniterative simulation can be important for obtaining starting distributions for iterative algorithms, as discussed in Gelman and Rubin (1993).

2. Normal-based Inference

2.1. Modes, Standard Errors, and the Normal Approximation

A point estimate and its associated standard error (or, more generally, its variance-covariance matrix), are motivated, explicitly or implicitly, by the normal approximation. Typically, the mean of the normal approximation is set equal to the mode (i.e., the maximum likelihood estimate or the posterior mode), and the inverse variance matrix is approximated by the negative of the second derivative matrix of the log posterior distribution at the mode. Computing these can be difficult in highly multivariate problems. Just finding the mode can require iteration, with Newton's method and EM (Dempster, Laird, and Rubin, 1977) being popular choices for common statistical models. Estimates of the "variance matrix" can be computed by analytic differentiation, numerical differentiation, or combined methods such as SEM (Meng and Rubin, 1991).

2.2. Approximation Using a Mixture of Normals

When the distribution is multimodal, it is necessary to run an iterative mode-finder several times, starting from different points, in an attempt to find all the modes. This strategy is also sensible and commonly used if the distribution is complicated enough that it *may* be multimodal. Once all the modes are found (possibly a difficult task) and the second derivative matrix calculated at each mode, the target distribution can be approximated by a mixture of k multivariate normals, each with its own mode μ_k and variance matrix Σ_k ; that is, the tar-

*Thanks to Donald B. Rubin for helpful comments and the National Science Foundation for financial support.

¹Another related and important topic, which we will not discuss here, is analytic approximations to integrals; see Tierney and Kadane (1986) and Morris (1988).

get density $P(x)$ is approximated by

$$\tilde{P}(x) = \sum_{k=1}^K \frac{\omega_k}{(2\pi)^{d/2} |\Sigma_k|^{1/2}} \exp\left(-\frac{1}{2}(x - \mu_k)^t \Sigma_k^{-1} (x - \mu_k)\right),$$

where K is the number of modes, d is the dimension of x , and ω_k is the mass of the k -th component of the multivariate normal mixture. The masses ω_k can be calculated by equating the approximate density \tilde{P} to the exact density P at the k modes, so that $\tilde{P}(\mu_k) = P(\mu_k)$, for $k = 1, \dots, K$. Assuming the modes are well-separated, this implies that for each k , the mass ω_k is roughly proportional to $|\Sigma_k|^{1/2} P(\mu_k)$.

2.3. Student- t Approximation

In general, one can replace the normal approximation by a multivariate t , with the same center and scale, but wider tails. A mixture of Student- t densities with η degrees of freedom has density

$$\tilde{P}(x) \propto \sum_{k=1}^K \frac{\omega_k}{|\Sigma_k|^{1/2}} (\eta + (x - \mu_k)^t \Sigma_k^{-1} (x - \mu_k))^{-(d+\eta)/2} \quad (2.1)$$

and can be simulated by first drawing from the normal mixture of Section 2.2 and then dividing the sampled vector by a χ_η^2 random deviate, divided by η . Because of its wide tails (and that it can be easily simulated and its density function is easy to calculate), the multivariate t will turn out to be useful as a starting distribution for the exact simulation methods described below.

3. Using Analytic Approximations and Importance Weights to Obtain Exact Simulations

3.1. Rejection Sampling

A simple way to draw samples from a target distribution P , using an approximate starting distribution P_0 , is *rejection sampling*, which requires the ability to calculate $P(x)/P_0(x)$, up to a proportionality constant, for all x . We will label $w(x) \propto P(x)/P_0(x)$, the *importance ratio* of x . In addition, rejection sampling requires a known constant M that is no less than $\sup w(x)$. The algorithm proceeds in two steps:

1. Sample x at random from $P_0(x)$.
2. With probability $\frac{w(x)}{M}$, reject x and return to step 1; otherwise, keep x .

An accepted x has the correct distribution $P(x)$; that is, the conditional distribution of drawn x , given it is accepted, is $P(x)$.

The above steps can be repeated to obtain additional independent samples from P . Rejection sampling cannot be used if no finite value of M exists, which will happen when P_0 has lighter tails than P , as when the support of P_0 is smaller than the support of P . (Hence the use of a multivariate t , instead of a normal, for a starting distribution.) In practice, when P_0 is not a good approximation to P , the required M will be so large that almost all samples obtained in step 1 will be rejected in step 2. The virtue of rejection sampling as an iterative simulation method is that it is self-monitoring—if the simulation is not working, you will know it, because no simulated draws will be accepted.

3.2. Approximate Rejection Sampling Using Importance Ratios

When no bound on $w(x)$ is known, rejection sampling is impossible. However, one can still draw samples $x^{(1)}, \dots, x^{(N)}$ from $P_0(x)$, and calculate their importance ratios, $w(x^{(l)})$, for $l = 1, \dots, N$, known only up to a proportionality constant. The method of *importance weighting* seeks to adjust the sampling by using $w(x^{(l)})$ to weight each random sample, $x^{(l)}$. Instead of discarding samples, those values $x^{(l)}$ with low importance ratios are just downweighted.

For any finite N , importance weighting gives only approximate results; it can thus be thought of as an iterative simulation method, improving as N increases. For importance weighting to be effective, the starting distribution P_0 should cover the target distribution P , in the sense that the importance ratios should not get too high. Even if importance ratios are unbounded, the method can still be useful—in contrast to rejection sampling—but the large values should be rare with respect to the target distribution.

Importance weights can be used to get a sequence of draws that approximately follow the target distribution by using the method of *importance resampling* (called SIR for “sampling-importance resampling” in Rubin, 1987, 1988). If N draws from the approximate starting distribution P_0 have been created, a sample of $n < N$ draws from a better approximation can be simulated as follows.

1. Sample a value x from the set $\{x^{(1)}, \dots, x^{(N)}\}$, where the probability of sampling each $x^{(l)}$ is proportional to the weight, $w(x^{(l)})$.
2. Sample a second value using the same procedure, but excluding the already-sampled value from the set.
3. Repeatedly sample without replacement $n - 2$ more times.

3.3. Iterative Importance Resampling

For any fixed N , importance resampling yields draws from an approximation to the target distribution. We can allow the approximation to improve in a smooth way as $n \rightarrow \infty$ by simply increasing N as n increases, that is, by expanding the pool of candidates $(x^{(1)}, \dots, x^{(N)})$ as more values are subsampled (without replacement) with probabilities proportional to their importance ratios; for example, N could increase as the square of n .

The simulation procedure thus becomes *iterative*: at each time $t = 1, 2, \dots$, a single draw x_t is taken from the set $(x^{(1)}, \dots, x^{(N)})$, with probabilities of sampling proportional to importance weights. The set $(x^{(1)}, \dots, x^{(N)})$ is created by supplementing the set of previously unsampled draws at time $t - 1$ with new independent draws from the approximate distribution P_0 . (We use the notation t for samples taken in succession, as opposed to n , the number of values in the final sample.)

If the importance ratios $P_0(x)/P(x)$ are bounded, the distribution of the samples x_t converge to the target distribution as $t \rightarrow \infty$.² The importance ratios will be bounded if the starting distribution P_0 has support at all the modes of interest in the target distribution P and has at least as heavy tails. We then say that the starting distribution is *overdispersed*, which is desirable.

The sequence x_1, x_2, \dots can be thought of as (dependent) draws from successively improving approximate distributions P_1, P_2, \dots that form a transition from the starting distribution P_0 toward the target distribution P . This is a conceptual improvement upon the basic attack of importance resampling, which provided no intermediate steps between the starting and target distributions. An obvious limitation, however, is that for all t , the supplemental draws are from P_0 , which may be a much less accurate approximation to P than that afforded by P_{t-1} . Section 4 reviews Markov chain methods, which modify the drawing distribution as t increases. Related ideas connecting importance weighting to iterative simulation appear in Tanner and Wong (1987), Gelfand and Smith (1990), and Kong, Liu, and Wong (1991).

²With unbounded importance ratios, the simulations may still converge to the target distribution. In general, the distributions of the resampled draws depends on the rate of increase of the population sample size N . Determining the necessary and sufficient conditions for convergence of importance resampling is a difficult problem not addressed in this paper.

4. A Review of Markov Chain Methods for Exact Simulation

4.1. Why is Markov Chain Simulation Needed?

Markov chain methods are especially desirable when no starting distribution is available that is accurate enough to produce useful importance weights. If the starting distribution is not close, the importance weights will be so variable that, for reasonable values of n and N , the set of draws from importance resampling will be a poor approximation to the target distribution. In order to correct the defects of the drawing distribution, P_0 , we must rely on a very large N .

In contrast, with any starting distribution that even loosely covers the target distribution, the steps of a Markov chain simulation directly improve the approximate distributions from which samples are drawn. Thus, the distributions, used for taking each draw, themselves converge to P as t increases. In a wide range of practical cases, it turns out that the iterations of a Markov chain simulation allow accurate inference from starting distributions that are much too vague for useful results from rejection or importance resampling. See Tierney (1991) for a unifying overview of many Markov simulation methods and Gelfand and Smith (1990) for an example in which importance resampling compares unfavorably to the Gibbs sampler.

4.2. The Method of Metropolis and its Generalizations

Given a target distribution $P(x)$, the generalized Metropolis algorithm (Hastings, 1970) draws a sequence of random points $(x^{(1)}, x^{(2)}, \dots)$ whose distributions converge to the target distribution. The sequence $(x^{(t)})$ may be considered a random walk whose stationary distribution is $P(x)$. The algorithm proceeds as follows:

1. Draw a starting point $x^{(0)}$, for which $P(x^{(0)}) > 0$, from a *starting distribution* $P_0(x)$.
2. For $t = 1, 2, \dots$:
 - (a) At iteration t , take as input the point $x^{(t-1)}$.
 - (b) Sample a candidate point \tilde{x} from a *jumping distribution* at time t , $J_t(\tilde{x}|x^{(t-1)})$.
 - (c) Calculate the ratio of importance ratios,

$$\tau = \frac{P(\tilde{x})}{P(x^{(t-1)})} \frac{J_t(x^{(t-1)}|\tilde{x})}{J_t(\tilde{x}|x^{(t-1)})}.$$

(τ is always defined, because a jump from $x^{(t-1)}$ to \tilde{x} can only occur if both $P(x^{(t-1)})$ and $J_t(\tilde{x}|x^{(t-1)})$ are nonzero.)

(d) Set

$$x^{(t)} = \begin{cases} \tilde{x} & \text{with probability } \min(r, 1) \\ x^{(t-1)} & \text{otherwise.} \end{cases}$$

This method requires the calculation of the relative importance ratios $P(x)/J_t(x|x')$ for all x, x' , and an ability to draw x from the jumping distribution $J_t(x|x')$ for all x' and t .

The proof that the iteration converges to the target distribution has two steps: first, it is shown that the simulated sequence $(x^{(t)})$ is a Markov chain with a unique stationary distribution, and second, it is shown that the stationary distribution equals the target distribution. The first step of the proof holds if the Markov chain is irreducible, aperiodic, and not transient (see, e.g., Feller, 1968). Except for trivial exceptions, the latter two conditions hold for a random walk on any proper distribution, and irreducibility holds as long as the random walk has a positive probability of eventually reaching any state from any other state; that is, the jumping distributions J_t must be able to eventually jump to all states with positive probability.

To see that the target distribution is the stationary distribution of the Markov chain generated by the generalized Metropolis algorithm, consider starting the algorithm at time $t-1$ with a draw $x^{(t-1)}$ from the target distribution $P(x)$. Now consider any two points y and z with positive probability under P , labeled so that $P(z)J_t(y|z) \geq P(y)J_t(z|y)$. The unconditional probability of a transition from y to z is

$$\Pr(x^{(t-1)}=y, x^{(t)}=z) = P(y)J_t(z|y),$$

and the unconditional probability of a transition from z to y is

$$\begin{aligned} \Pr(x^{(t-1)}=z, x^{(t)}=y) &= P(z)J_t(y|z) \frac{P(y)J_t(z|y)}{P(z)J_t(y|z)} \\ &= P(y)J_t(z|y), \end{aligned}$$

which is the same as the probability of a transition from y to z . Since their joint distribution is exchangeable, $x^{(t)}$ and $x^{(t-1)}$ have the same marginal distributions, and so P is the stationary distribution of the Markov chain.

The method of Metropolis et al. (1953) is the same as that described above, with the restrictions that the jumping distribution be symmetric and not depend on t : $J_t(y|z) = J_t(z|y) = J_0(z|y)$ for any y, z .³

³Barker (1965) suggests a method identical to Metropolis', except that the switching probability at each step is changed from $\min(r, 1)$ to $\frac{r}{1+r} = \frac{P(\tilde{x})}{P(\tilde{x}) + P(x^{(t-1)})}$. Alternatively, Barker's method may be considered a generalized Metropolis algorithm in

4.3. Gibbs Sampling

Geman and Geman (1984) introduced "Gibbs sampling," a procedure for simulating a multivariate probability distribution $P(x) = P(x_1, \dots, x_d)$, by performing a random walk on the vector $x = (x_1, \dots, x_d)$, altering one, possibly vector, component x_i at a time. At iteration t , an ordering of the d components of x is chosen and, in turn, each $x_i^{(t)}$ is sampled from the conditional distribution given that all the other components remained fixed:

$$P(x_i | x_{-i}^{(t-1)}),$$

where $x_{-i} = (x_1, \dots, x_{i-1}, x_{i+1}, \dots, x_d)$. Each of these d steps can be interpreted as one iteration of the generalized Metropolis algorithm, with the following jumping distribution, which only jumps along the i -th component, and does so with the conditional probability of x_i given $x_{-i} = x_{-i}^{(t-1)}$ obtained from the target distribution:

$$J_{it}[\text{Gibbs}](\tilde{x} | x^{(t-1)}) = \begin{cases} P(\tilde{x}_i | x_{-i}^{(t-1)}) & \text{if } \tilde{x}_{-i} = x_{-i}^{(t-1)} \\ 0 & \text{otherwise.} \end{cases}$$

Under this jumping distribution, the ratio of importance ratios is

$$\begin{aligned} r &= \frac{P(\tilde{x})}{P(x^{(t-1)})} \frac{J_{it}(x^{(t-1)} | \tilde{x})}{J_{it}(\tilde{x} | x^{(t-1)})} \\ &= \frac{P(\tilde{x})}{P(x^{(t-1)})} \frac{P(x_i^{(t-1)} | x_{-i}^{(t-1)})}{P(\tilde{x}_i | x_{-i}^{(t-1)})} \\ &\equiv 1, \end{aligned}$$

and so jumps always occur, as prescribed by the Gibbs sampler. Obviously, as described, the Gibbs sampler requires the ability to draw from the conditional distributions derived from the target distribution.

Usually, one iteration of the Gibbs sampler is defined as above, to include all d Metropolis steps corresponding to the d components of x , thereby updating all of x at each iteration. It is possible, however, to define Gibbs sampling without the restriction that each component be updated in each iteration.

4.4. Gibbs Sampling with Approximations

For some problems, sampling from some, or all, of the conditional distributions $P(x_i | x_{-i})$ is impossible, and one must resort to approximations $g(x_i | x_{-i})$. Trying

which the jumping distribution $J(y|z)$ is replaced by

$$J[\text{Barker}](y|z) = J(y|z) \frac{P(y)}{P(y) + P(z)} \quad \text{for all } y \neq z.$$

See also Hastings (1970) for further discussion of Barker's, Metropolis', and related algorithms.

to perform the Gibbs sampler directly, using the conditional distributions g instead of P , will not work. The generalized Metropolis algorithm, however, is suited for the task. As in the Gibbs sampler, we must choose an ordering for altering the d elements of x ; the jumping function at the i -th Metropolis step at iteration t is then

$$J_{it}(\tilde{x}|x^{(t-1)}) = \begin{cases} g(\tilde{x}_i|x_{-i}^{(t-1)}) & \text{if } \tilde{x}_{-i} = x_{-i}^{(t-1)} \\ 0 & \text{otherwise,} \end{cases}$$

and the ratio of importance ratios is

$$\begin{aligned} r &= \frac{P(\tilde{x})}{P(x^{(t-1)})} \frac{J_{it}(x^{(t-1)}|\tilde{x})}{J_{it}(\tilde{x}|x^{(t-1)})} \\ &= \frac{P(\tilde{x}_i|x_{-i}^{(t-1)})}{P(x_i^{(t-1)}|x_{-i}^{(t-1)})} \frac{g(x_i^{(t-1)}|x_{-i}^{(t-1)})}{g(\tilde{x}_i|x_{-i}^{(t-1)})}, \end{aligned}$$

which is identically equal to 1 only if $g(x|x_{-i}) \equiv P(x|x_{-i})$. If g is only an approximation, the Metropolis step will have a positive probability of not jumping.

5. Discussion

A large and expanding family of iterative and non-iterative simulation algorithms exist for approximating a target distribution using samples from a starting distribution. Despite the non-iterative appearance of rejection and importance sampling, all the available methods (except for direct simulation) yield exact simulations of the target distribution only in the limit that the number of samples $n \rightarrow \infty$. (Rejection sampling, however, has the advantage that once samples have been obtained, they are known to follow the target distribution.) In every approach, the starting distribution is key; an overdispersed start has long been recognized as necessary for rejection and importance sampling, and more recently been advocated for Markov chain simulation (Gelman and Rubin, 1992, 1993).

Monitoring convergence of all these methods (except for rejection sampling) can be difficult for any of these algorithms in practice. Gelman and Rubin (1993) present one approach based on performing multiple independent simulation runs which, while designed for iterative simulation methods such as the Gibbs sampler, might also be useful for inference from importance sampling.

6. References

- Barker, A. A. (1965), "Monte Carlo calculations of radial distribution functions for a proton-electron plasma", *Aust. J. Phys.*, **18**, 119-133.
- Besag, J., and Green, P. J. (1993), "Spatial statistics and Bayesian computation", *J. Roy. Stat. Soc. B*, **55**, to appear.
- Dempster, A. P., Laird, N. M., and Rubin, D. B. (1977), "Maximum likelihood from incomplete data via the EM algorithm (with discussion)", *J. Roy. Stat. Soc. B*, **39**, 1-38.
- Feller, W. (1968), *An Introduction to Probability Theory and Its Applications*. New York: Wiley.
- Gelfand, A. E., and Smith, A. F. M. (1990), "Sampling-based approaches to calculating marginal densities", *J. Amer. Stat. Assoc.*, **85**, 398-409.
- Gelfand, A. E., Hills, S. E., Racine-Poon, A., and Smith, A. F. M. (1990), "Illustration of Bayesian inference in normal data models using Gibbs sampling", *J. Amer. Stat. Assoc.*, **85**, 398-409.
- Gelman, A., and Rubin, D. B. (1992), "A single series from the Gibbs sampler provides a false sense of security", in *Bayesian Statistics 4*, ed. J. Bernardo, Oxford University Press.
- Gelman, A., and Rubin, D. B. (1993), "Inference from iterative simulation using multiple sequences", *Stat. Sci.*, to appear.
- Geman, S., and Geman, D. (1984), "Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images", *IEEE Trans. on Pattern Analysis and Machine Intelligence*, **6**, 721-741.
- Gilks, W. R., Clayton, D. G., Spiegelhalter, D. J., Best, N. G., McNeil, A. J., Sharples, L. D., and Kirby, A. J. (1993), "Modelling complexity: applications of Gibbs sampling in medicine", *J. Roy. Stat. Soc. B*, **55**, to appear.
- Hastings, W. K. (1970), "Monte-Carlo sampling methods using Markov chains and their applications", *Biometrika*, **57**, 97-109.
- Kong, A., Liu, J., and Wong, W. H. (1991), "Sequential imputations and Bayesian missing data problems", Technical Report #321, Department of Statistics, University of Chicago.
- Meng, X. L., and Rubin, D. B. (1991), "Using EM to obtain asymptotic variance-covariance matrices: the SEM algorithm", *J. Amer. Stat. Assoc.*, **86**, 899-909.
- Metropolis, N., Rosenbluth, A. W., Rosenbluth, M. N., Teller, A. H., and Teller, E. (1953), "Equation of state calculations by fast computing machines", *J. Chem. Phys.*, **21**, 1087-1092.

- Metropolis, N., and Ulam, S. (1949), "The Monte Carlo method", *J. Amer. Stat. Assoc.*, 44, 335-341.
- Morris, C. N. (1988), "Approximating posterior distributions and posterior moments", in *Bayesian Statistics 3*, 327-344, ed. J. Bernardo, Oxford University Press.
- Rubin, D. B. (1987), "A noniterative sampling/importance resampling alternative to the data augmentation algorithm for creating a few imputations when fractions of missing information are modest: the SIR algorithm", *J. Amer. Stat. Assoc.*, 82, 543-546.
- Rubin, D. B. (1988), "Using the SIR algorithm to simulate posterior distributions", in *Bayesian Statistics 3*, 395-402, ed. J. Bernardo, Oxford University Press.
- Smith, A. F. M., and Roberts, G. O. (1993), "Bayesian computation via the Gibbs sampler and related Markov chain Monte Carlo methods", *J. Roy. Stat. Soc. B*, 55, to appear.
- Tanner, M. A., and Wong, W. H. (1987), "The calculation of posterior distributions by data augmentation (with discussion)", *J. Amer. Stat. Assoc.*, 82, 528-550.
- Tierney, L. (1991), "Exploring posterior distributions using Markov chains", *Proceedings of the Interface Conference*.
- Tierney, L., and Kadane, J. B. (1986), "Accurate approximations for posterior moments and marginal densities", *J. Amer. Stat. Assoc.*, 81, 82-86.

Software for the Gibbs sampler

W. R. Gilks, A. Thomas and D. J. Spiegelhalter
Medical Research Council Biostatistics Unit, Cambridge, UK

Abstract

Gibbs sampling has enormous potential for analysing complex datasets. However routine use of Gibbs sampling has been hampered by the lack of general purpose software for its implementation. Until now all applications have involved writing one-off computer code in low or intermediate-level languages such as *C* or *FORTRAN*.

We describe some general purpose software we are currently developing for implementing Gibbs sampling: *BUGS* (Bayesian inference using Gibbs sampling). The *BUGS* system comprises three components. First, a natural language for specifying complex models; second, an 'expert system' for deciding appropriate methods for obtaining samples required by the Gibbs sampler; and third, a sampling module containing numerical routines to perform the sampling. *S* objects are used for data input and output. *BUGS* is written in *Modula-2*, and runs under both *DOS* and *UNIX*.

1 Introduction

Gibbs sampling (Geman and Geman (1984), Hastings (1970)) is a technique for simulating samples from the joint posterior distribution of the unknown quantities in a statistical model. It has been shown to have enormous potential for the statistical analysis of complex data sets (see, for example, Gelfand and Smith (1990), Gelfand *et al* (1990), Smith and Roberts (1992), Gilks *et al* (1992b)). Gibbs sampling succeeds because it reduces the problem of

dealing simultaneously with a large number of intricately related, unknown parameters and missing data, into a much simpler problem of dealing with one unknown quantity at a time, sampling each from its *full conditional* distribution (see Section 5). Methods of sampling from complicated full conditionals include adaptive rejection sampling (Gilks and Wild (1992); Gilks (1992); Gilks *et al* (1992a)) and the ratio-of-uniforms method (Wakefield *et al* (1992)). We pass over a description of the basic methodology of Gibbs sampling, this having been presented already at this meeting (Gelman (1992)).

Notwithstanding the growing popularity of Gibbs sampling, its routine use has been hampered by a lack of general purpose software for its implementation. Until now all applications have involved writing one-off computer code in low or intermediate-level languages such as *C* or *FORTRAN*. In our own experience, writing and debugging a Gibbs sampler for a moderately complex application can take anything from a few days to several weeks. Thereafter, modifying the program (for example to elaborate the model or to apply it to different data) might take several hours. This compares dismally with model fitting in *GLIM* or *S*, for which model specification may take just a few minutes, and modifications may take just a few seconds. The fact that increasing numbers of statisticians are expending considerable effort to use Gibbs sampling clearly indicates that currently available software is inadequate in many applications. It is our aim to make Gibbs sampling as easy to

implement as generalised linear models in *GLIM* or *S*. To this end we are developing *BUGS* (Bayesian inference using Gibbs sampling). In this paper we describe our approach.

The main requirements of *BUGS* are that it should

- accomodate a very large class of models;
- enable models to be specified concisely;
- construct and sample from full conditional distributions automatically.

We discuss these requirements in the following Sections.

2 Class of models

The class of models we would like to be able to handle in *BUGS* is extensive. This class should include most if not all of the models that have been applied in the Gibbs sampling literature, including cluster analysis models (Gilks *et al* (1989)); survival analysis models (Clayton (1991)); image analysis models (Besag *et al* (1991)); econometric models (Blattberg and George (1991)); generalised linear random-effects models (Zeger and Karim (1991), Dellaportas and Smith (1992)); complex genetic models (Sheehan and Thomas (1992), Thompson and Guo (1992), Thomas (1992)); disease mapping models (Bernardinelli and Montomoli (1992), Clayton and Bernardinelli (1992)); hidden Markov models (Kirby (1992)); change-point models (Carlin *et al* (1992)); non-linear pharmacokinetic models (Wakefield *et al* (1991), Berzuini *et al* (1992)); and constrained data models (Gelfand *et al* (1992)). Moreover, Gilks *et al* (1992b) point out that many applications comprise several linked submodels.

Clearly we cannot hope to define the class of models of interest in a tight al-

gebraic form, as in other statistical modelling software. The only common feature of the applications listed above is that models are specified through a series of model components or submodels. To focus the following discussion, we now briefly present one such application.

An example

Gilks and Richardson (1992) describe an application of Gibbs sampling in occupational epidemiology. In this application the aim is to estimate disease risks associated with chronic exposure to industrial agents (chemicals, dust or fibres). A *disease study* has been conducted in which each individual's status for a particular disease has been recorded, but his exposure status on each of several agents is unknown. In another study (the *exposure study*) on different individuals, exposure statuses have been recorded but disease statuses are unknown for each individual. Thus we have the usual ingredients for a logistic regression model: independent variables (exposure statuses) and a dependent variable (disease status); but unusually we know both of these for no-one. The vital link between the two sets of individuals is *occupation*: we know the job-description of every individual.

For the disease-study individuals Gilks and Richardson (1992) propose the following model:

$$D_i \sim \text{Bernoulli}(\theta_i) \quad (1)$$

where D_i is the disease status (0 = not diseased; 1 = diseased) for the i^{th} individual, θ_i is his probability of disease and

$$\theta_i = \text{logit}^{-1}(\beta_0 + \sum_k \beta_k E_{ik}), \quad (2)$$

where $\text{logit}^{-1}(x)$ denotes the inverse logit function $1/(1 + \exp(-x))$ and E_{ik} is the (unobserved) exposure status (0 = unexposed; 1 = exposed) of individual i to

agent k . Equations (1) and (2) together define a classical logistic regression model.

For the exposure-study individuals, Gilks and Richardson (1992) propose:

$$m_{jk} \sim \text{Binomial}(\pi_{jk}, n_j) \quad (3)$$

where n_j is the number of exposure-study individuals in job type j ; m_{jk} is the number of these who were exposed to agent k ; and π_{jk} is an exposure probability. The set $\{m_{jk}, n_j\}$ comprise a *job-exposure matrix*.

The link between the two studies is then provided by

$$E_{ik} \sim \text{Bernoulli}(\pi_{j(i),k}) \quad (4)$$

where $j(i)$ denotes the job type of disease-study individual i .

The Bayesian model specification is completed with priors

$$\beta_k \sim \text{Normal}(\mu_k, \sigma_k^2), \quad (5)$$

and

$$\pi_{jk} = \text{logit}^{-1} \phi_{jk} \quad (6)$$

with

$$\phi_{jk} \sim \text{Normal}(\mu_k^*, \sigma_k^{*2}). \quad (7)$$

Directed acyclic graphical models

The model set out in submodels (1)–(7) incompletely specifies the joint distribution of the model parameters and data. These submodels merely tie down a few conditional marginal distributions. However, in an intuitive sense, model (1)–(7) is already complete: if further structure had been required it could have been specified explicitly. Thus to complete the joint distribution of the parameters and data, one seeks some kind of assumption of ‘independence’ between the submodels. This is provided by the *directed Markov assumption* (Lauritzen et al (1990)), which simply states that the joint distribution of all the model parameters and data is given by the product of all the submodels. Thus the joint

distribution from (1)–(7) is:

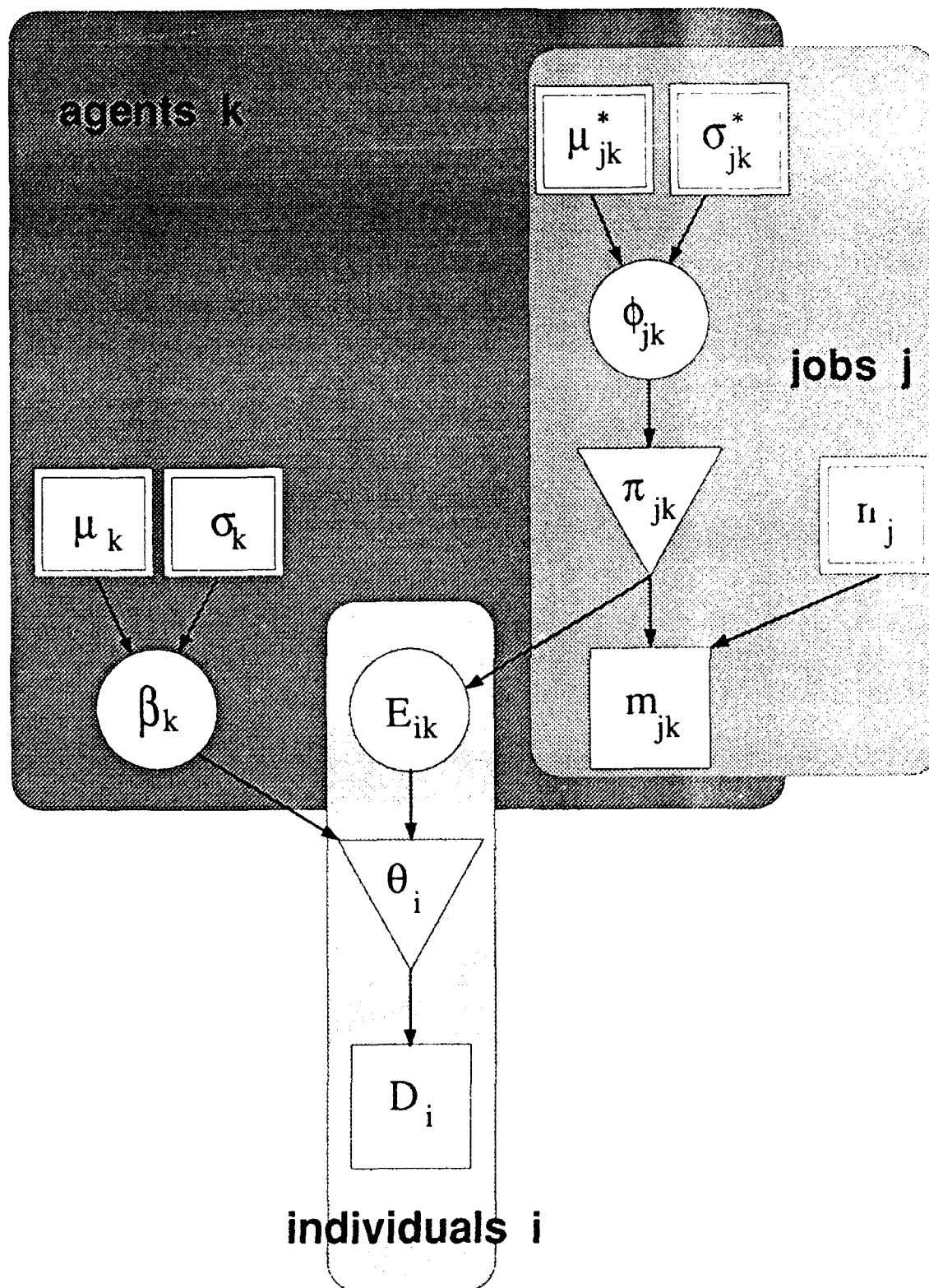
$$\begin{aligned} & \prod_i p(D_i | \theta_i) \prod_k p(\beta_k | \mu_k, \sigma_k) \\ & \times \prod_{ik} p(E_{ik} | \pi_{j(i),k}) \prod_{jk} p(m_{jk} | \pi_{jk}, n_j) \\ & \times \prod_{jk} p(\phi_{jk} | \mu_k^*, \sigma_k^{*2}) \quad (8) \end{aligned}$$

where deterministic equations (2) and (6) are assumed to hold.

That the directed Markov assumption is natural for model (1)–(7) is most easily seen from a *graph* of the model (Figure 1), in which round nodes denote unobserved stochastic variables (model parameters or missing exposures); square nodes with a single border denote observed stochastic variables (i.e. data); square nodes with a double border denote fixed quantities in prior distributions; triangles denote deterministic variables; and *edges* (arrows) denote dependencies specified in the submodels. Figure 1 is a *directed acyclic graph* (Whittaker (1990)), since all the edges are directed and it is not possible, just by following the directions of the edges, to return to a node after leaving it. Thus there exists a partial ordering of the nodes, and the directed Markov assumption is seen simply to be a natural Markov-type assumption on the partial ordering.

We will refer to the node on the left of a submodel (e.g. D_i in (1)) as a *child* of the nodes on the right of that submodel, and the nodes on the right of the submodel as the *parents* of the node on the left. A *stochastic parent* of a given node is a stochastic node which is an ancestor (in an obvious extension of terminology) of the given node, where only deterministic nodes intervene between the two. Likewise, a *stochastic child* of a given node is a stochastic node which is a descendant of the given node where only deterministic nodes intervene between the two. For example, from Figure 1, θ_i is the only parent of D_i , and D_i is the child of θ_i ; E_{ik} is one of the stochastic parents of D_i , and D_i is

Figure 1. Graphical representation of model (1) - (7).



the stochastic child of E_{ik} .

We will refer to a set of submodels (such as (1)–(7)) corresponding to a directed acyclic graph, together with the directed Markov assumption, as a *directed acyclic graphical (DAG) model*. DAG models provide a very rich class of models which are particularly amenable to estimation by Gibbs sampling (see Section 5). Within this class we would like the family of algebraic and stochastic forms for the individual submodels to be very general, accommodating non-linearity and all commonly used statistical distributions. Thus we have developed a user interface and corresponding facilitating algorithms, as we describe below.

3 Model specification

It is tempting to try to devise a graphical user interface to *BUGS*, allowing models to be specified through drawing graphs on the screen. However, a graph gives only structural information: the precise form of each submodel would still need to be specified. Moreover, a graph may not even give full structural information. For example in Figure 1 the structure of the mapping of jobs onto individuals $j(i)$ is suppressed. To express the full model structure in the form of a graph on the screen would entail drawing one node for each of possibly several thousand variables in the model. Thus a graph, whilst of conceptual value, is of limited practical value for model specification.

Instead we have developed a semantic interface to *BUGS* which mimics as far as possible the natural model specification, as set out for example in equations (1)–(7). Thus the stochastic relationship (1) is specified by:

$$D[i] \sim \text{Bernoulli}(\text{theta}[i]); \quad (9)$$

and the deterministic relationship (2) is specified by:

$$\text{Logit}(\text{theta}[i]) \leftarrow \text{beta}[nA1] +$$

$$\text{InProd}(\text{beta}[1:nA], E[i,]); \quad (10)$$

Each of these model statements *declares* the node on the left of the expression. The \sim in (9) declares $D[i]$ to be a stochastic node, and the \leftarrow in (10) declares $\text{theta}[i]$ to be a deterministic node. In (10), *InProd* denotes an inner product; $\text{beta}[1:nA]$ denotes $\beta_1, \beta_2, \dots, \beta_{nA}$; and $E[i,]$ denotes E_1, E_2, \dots, E_{nA} (where nA is the number of agents in this example). The full *BUGS* specification of model (1)–(7) is given in Figure 2.

Explicit *for* loops and implicit loops as in (10) enable models to be specified succinctly. Also, to aid succinctness, some of the structural details of the model can be suppressed in the model specification, instead being read into *BUGS* as data. For example, the occupation of each individual in the disease study appears in Figure 2 only in the form $J[i]$; and $J[i]$ must be read in as data. (*BUGS* deduces this since $J[i]$ does not appear on the left of any declaration.)

In principle, the Gibbs sampler easily handles missing data by treating them as unknown model parameters. Since there could be many haphazardly scattered missing data in a given application, it would be rather tedious to have to identify them through individual model statements. To avoid this, *BUGS* adopts the convention that any stochastic variable declared on the left hand side of a model statement, but which is not found in the *data* file, is to be treated as a free model parameter in the Gibbs sampling and updated at each iteration. Any variables found in the *data* file are assumed known, and will not be updated.

It is important to understand that model specification statements in *BUGS* are *declarative*; that is, they are not instructions to perform calculations *per se*; rather, they are declarations: specifying the type of each node and its relationships with other nodes. Indeed, in the Gibbs sampling, $D[i]$ is never actually *sampled*

Figure 2. Model specification in *BUGS*.

```

model Jobs;
  data in "bugs\dat\Jobs.dat";
  inits in "bugs\in\Jobs.in";
const
  nI = 1000, # number of individs
  nA = 4, # number of agents
  nJ = 2, # number of jobs
  nA1 = nA + 1;
var
  D[nI], J[nI], theta[nI], E[nI, nA], pi[nJ, nA],
  phi[nJ, nA], mu_star[nJ, nA], tau_star[nJ, nA],
  m[nJ, nA], n[nA], beta[nA1], mu[nA1], tau[nA1];
{
  for (i in 1:nI)
  {
    # disease model
    D[i] ~ Bernoulli(theta[i]);
    Logit(theta[i]) <- beta[nA1] + InProd(beta[1:nA], E[i,]);

    for (k in 1:nA)
    {
      # exposure model
      E[i, k] ~ Bernoulli(pi[J[i], k]);
    }
  }

  for (j in 1:nJ)
  {
    for (k in 1:nA)
    {
      # measurement model
      m[j, k] ~ Binomial(pi[j, k], n[j]);
      Logit(pi[j, k]) <- phi[j, k];
      phi[j, k] ~ Normal(mu_star[j, k], tau_star[j, k]);
    }
  }

  for (k in 1:nA1)
  {
    beta[k] ~ Normal(mu[k], tau[k]);
  }
}

```

from the Bernoulli distribution specified in (9), since $D[i]$ is fixed. Thus the order of model specification statements in *BUGS* is, to a considerable extent, arbitrary. For example, the *for* loops in Figure 2 could have been specified in a different order, perhaps starting with the measurement model loop. Equally, the order of statements within the inner loops in Figure 2 is arbitrary.

4 Structure of *BUGS*

BUGS comprises three distinct modules: the parser, the code generator and the Gibbs sampler.

The parser

The parser is responsible for reading the user's code in the specification file and assimilating node declarations and dependencies. For example, in Figure 2, node $E[i, k]$ is set up as a stochastic node having a Bernoulli distribution, and a pointer to node $pi[J[i], k]$ is stored at node $E[i, k]$. Thus, the parser builds up the partial ordering (the *node tree*) representing the full structure of the graphical model. For deterministic nodes, such as that defining $\theta[i]$ in Figure 2, the functional form of the deterministic relationship is stored in the form of a stack associated with the node, containing instructions in *reverse Polish* notation. For example, the expression: $b * x + c$ would be stored as a stack: $b, x, *, c, +$; which will be interpreted as: "push b ; push x ; pop b and x ; push $b * x$; push c ; pop $b * x$ and c ; push $b * x + c$ ".

The parser reads data from the *data* file specified by the user, and stores these data at the relevant nodes. The parser also discerns which nodes are to be updated in Gibbs sampling and which are to remain fixed, as described in Section 3. Initial values for model parameters (and missing data) are obtained from the user-defined *inits* file. Any free model parameter not

found in the *inits* file is assigned an initial value by forwards sampling, provided the user did not assign it an improper prior. Currently, all data and initial values read into *BUGS* must be stored in the input files in the form of *S* objects.

A number of consistency checks are carried out by the parser, for example to ensure that all nodes not appearing on the left hand side of a model statement are assigned values in the *data* file; and to check that each stochastic and deterministic function has the right number of arguments, of the right type.

The code generator

The code generator fills out the node tree constructed by the parser, to store at each node pointers to its stochastic children (see Section 2). For example, in the example in Figure 2, pointers to nodes $E[i, k]$ for all individuals i in job j are stored at node $\phi[j, k]$.

Having filled out the node tree, the code generator uses a small expert system to decide, for each node, how best to sample from its full conditional distribution in the Gibbs sampling module (see Section 5).

The code generator generates a low-level language which is output to a system file. The contents of this file are intelligible to the user, although it should not be necessary for the user to concern himself with this. The low-level language describes all the attributes of each node, including references to parents and stochastic children, as well as an indicator of the method to be used for sampling from its full conditional distribution. In this low-level language, subscripts are not used: each node has equal status with all other nodes, regardless of whether it derived originally from a subscripted variable.

The Gibbs sampler

The Gibbs sampler operates on the low-level code produced by the code genera-

tor. In principle, the Gibbs sampler could be run on a machine other than that which produced the low-level code, since the low-level code can be read from file. The Gibbs sampling module interprets the low-level code (in particular the reverse Polish of deterministic nodes); constructs appropriate quantities relating to full conditional distributions (see Section 5); samples from full conditionals; controls the order in which nodes are resampled; and prints results to a file in the form of *S* objects.

5 Full conditional distributions

Construction of full conditionals

The *full conditional distribution* of a stochastic node is its conditional distribution conditioning on the current values of all other stochastic nodes in the graph. For a DAG model, the full conditional distribution for a given stochastic node is proportional to the product of the submodel declaring the given node with the submodels declaring the stochastic children of the given node (see Section 2). For example, the full conditional for β_1 in model (1)–(7) is proportional to:

$$p(\beta_1 | \mu_1, \sigma_1) \prod_i p(D_i | \theta_i) \quad (11)$$

where deterministic equation (2) holds. The proportionality constant, required to make the full conditional integrate to unity, will be a function of nodes other than β_1 . However, methods of sampling from full conditionals used in *BUGS* (see below) do not require explicit evaluation of integration constants.

BUGS also has a mechanism for dealing with models which are not DAG models, for example image analysis models (Besag et al (1991)) and disease mapping models (Bernardinelli and Montomoli (1992), Clayton and Bernardinelli (1992)). There

is no obvious rule here, analogous to the directed Markov assumption, for obtaining joint distributions from user-specified submodels. Moreover, it is easy to construct sets of submodels for which no consistent joint distribution exists. The rule used by *BUGS* in constructing the full conditional distribution for a given node in a non-DAG model is to form the product of the submodel declaring the given node with the submodels declaring the stochastic children of the given node, *excluding* any submodels declaring stochastic children which are also stochastic parents of the given node. This rule may not always make sense, but it allows *BUGS* to accommodate the above non-DAG models and would allow, for example, the full conditional for a given node to be supplied *explicitly* by the user.

Note that in (11), the full conditional for β_1 involves only a subset of the other nodes in the graph. This is generally true for DAG models, and is important for computational efficiency. The ingredients required for constructing a full conditional for a given node are all identified explicitly through the list of pointers at that node in the low-level language produced by the code generator.

Sampling from full conditionals

The code generator contains a small expert system for deciding the best method of sampling from full conditionals. Currently, the first choice is to identify conjugacy (where the full conditional reduces analytically to a well-known distribution) and sample accordingly. For example, if $y \sim N(\mu, \sigma^2)$ and $\mu \sim N(\nu, \tau^2)$, then the full conditional for μ will be $N(\frac{\nu\tau^2 + y\sigma^2}{\tau^2 + \sigma^2}, \frac{1}{\sigma^{-2} + \tau^{-2}})$. The expert system recognises a small number of canonical cases of conjugacy, and also recognises conjugacy in situations which reduce to a canonical case through linear transforms. For example if μ is replaced with a lin-

ear form $a + b\mu$, then the full conditional for μ still reduces through conjugacy (to $N(\frac{b(y-a)\tau^2 + \nu\sigma^2}{b^2\tau^2 + \sigma^2}, \frac{1}{b^2\tau^2 + \sigma^2})$).

The method of second choice for sampling from full conditionals is adaptive rejection sampling. The original method of adaptive rejection sampling (Gilks and Wild (1992)) requires the density to be log-concave ($\frac{d^2 \log f(x)}{dx^2} \leq 0$) and the user to supply subroutines to calculate derivatives of the log density. A second version (Gilks (1992)) does not use derivatives but still requires log-concavity. The most recent version (Gilks et al (1992)) accommodates non-log-concavity through use of a generalised rejection function and a post-acceptance Hastings-Metropolis step (Hastings (1970)).

6 Discussion

The *BUGS* program is still under development. It is our hope to have a basic beta-test version working within three months of this meeting. Beyond that we have several plans for further development. In particular we plan to allow input of data which are not S objects; to provide an interface to enable key parameters to be monitored during the Gibbs sampling; and to allow the user to control which parameters are output. We have no immediate plans to incorporate convergence diagnostics into *BUGS*, as this field is still evolving rapidly and most convergence diagnostics already proposed can be applied post-hoc.

References

- Bernardinelli, L. and Montomoli C. (1992) Empirical Bayes vs fully Bayesian analysis of geographical variation in disease risk. *Statistics in Medicine*, (in press).
- Berzuini, C., Spiegelhalter, D. J. and Bellazzi, R. (1992) Bayesian networks in patient monitoring. *Artificial Intelligence in Medicine*, (submitted).
- Besag, J., York, J., and Mollie, A. (1991) Bayesian image restoration, with applications in spatial statistics (with discussion). *Ann. Inst. Statist. Math.*, **43**, 1-59.
- Blattberg, R. C. and George, E. I. Shrinkage estimation of price and promotional elasticities: seemingly unrelated equations. (1991) *J. Amer. Statist. Assoc.*, **86**, 304-315.
- Carlin, B. P., Gelfand, A. E., Smith, A. F. M. (1992) Hierarchical Bayesian analysis of changepoint problems. *Applied Statistics*, **41**, 389-405.
- Clayton, D. G. (1991) A Monte Carlo method for Bayesian inference in frailty models *Biometrics*, **47**, 467-485.
- Clayton, D. G. and Bernardinelli, L. (1992) Bayesian methods for mapping disease risk. In: *Small Area Studies in Geographical and Environmental Epidemiology*, (Eds. Cuzick, J. and Elliott, P.), Oxford University Press, (to appear).
- Dellaportas, P. and Smith, A. F. M. (1992) Bayesian inference for generalised linear models and proportional hazards models via Gibbs sampling. *Applied Statistics*, (submitted).
- Gelfand, A. E., Hills, S. E., Racine-Poon, A. and Smith, A. F. M. (1990) Illustration of Bayesian inference in normal data models using Gibbs sampling. *Journal of the American Statistical Association*, **85**, 972-985.
- Gelfand, A. E. and Smith, A. F. M. (1990) Sampling based approaches to calculating marginal densities. *Journal of the American Statistical Association*, **85**, 398-409.
- Gelfand, A. E., Smith, A. F. M. and Lee, T. M. (1992) Bayesian analysis of constrained parameter and truncated data problems. *J. Amer. Statist. Assoc.*, (in press)
- Gelman A. (1992) This volume.
- Geman, S. and Geman, D. (1984) Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images'. *IEEE*

- Transactions on Pattern Analysis and Machine Intelligence*, PAMI-6, 721-741.
- Gilks, W. R., Oldfield, L. and Rutherford, A. (1989). Statistical analysis. In *Leucocyte Typing IV: White Cell Differentiation Antigens* (eds. W. Knapp, B. Dorken, W. R. Gilks et al), pp6-12. Oxford University Press: Oxford.
- Gilks, W. R. (1992) Derivative-free adaptive rejection sampling for Gibbs sampling. *Bayesian Statistics 4*, (eds: Bernardo, J., Berger, J., Dawid, A. P. and Smith, A. F. M.), (in press).
- Gilks, W. R., Best, N. G. and Tan, K. K. C. (1992a) *Adaptive rejection sampling from non log-concave densities*. Technical report, Medical Research Council Biostatistics Unit, Cambridge (in preparation).
- Gilks, W. R., Clayton, D. G., Spiegelhalter, D. J., Best, N. G., McNeil, A. J., Sharples, L. D. and Kirby, A. J. (1992b) Modelling complexity: applications of Gibbs sampling in medicine. *J. Roy. Statist. Soc. Ser. B* (to appear).
- Gilks, W. R. and Richardson, S. (1992b) Analysis of disease risks using ancillary risk factors, with application to job-exposure matrices. *Statistics in Medicine*, (to appear).
- Gilks, W. R. and Wild, P. (1992) Adaptive rejection sampling for Gibbs sampling. *Applied Statistics*, 41, 337-348.
- Hastings, W.K. (1970). Monte Carlo sampling methods using Markov chains and their applications. *Biometrika*, 57, 97-109.
- Kirby, A. J. (1992) *Statistical modelling for the precursors of cervical cancer*. PhD Thesis, University of Cambridge.
- Lauritzen, S. L., Dawid, A. P., Larsen, B. N. and Leimer, H. G. (1990) Independence properties of directed Markov fields. *Networks*, 20, 491-505.
- Sheehan, N. and Thomas, A. (1992) On the irreducibility of a Markov chain defined on a space of genotype configurations by a sampling scheme. *Biometrics* (in press).
- Smith, A. F. M. and Roberts, G. O. (1992) Bayesian computation via the Gibbs sampler and related Markov chain Monte Carlo methods. *J. Roy. Statist. Soc. Ser. B* (to appear).
- Thomas, D. C. (1992) Fitting genetic data using Gibbs sampling: an application to nevus counts in 38 Utah kindreds. *Cytogenetics and Cell Genetics*, (in press).
- Thompson, E. A. and Guo, S. W. (1992) Evaluation of likelihood ratios for complex genetic models. *IMA Journal of Mathematics Applied in Medicine and Biology*, (in press).
- Wakefield, J., Gelfand, A. E. and Smith, A. F. M. (1992) Efficient generation of random variates via the ratio-of-uniforms method. *Statistics and Computing*, 1, (in press).
- Wakefield J. C., Smith, A. F. M., Racine-Poon, A. and Gelfand, A. E. (1991) Bayesian analysis of linear and non-linear population models using the Gibbs sampler. Technical Report, Imperial College, London University.
- Whittaker, J. (1990) *Graphical models in applied multivariate statistics*. John Wiley and Sons: New York.
- Zeger, S. L. and Karim, M. R. (1991) Generalised linear models with random effects; a Gibbs sampling approach. *J. Amer. Statist. Assoc.*, 86, 79-86.

Bayesian inference by simulation in a stochastic model from hematology

Michael A. Newton

Departments of Statistics and Biostatistics
University of Wisconsin-Madison
Madison, WI 53706
Internet: newton@stat.wisc.edu

Peter Guttorp

Department of Statistics
University of Washington
Seattle, WA 98195

Janis L. Abkowitz

Division of Hematology
Department of Medicine
University of Washington
Seattle, WA 98195

ABSTRACT

A particular Markov chain Monte Carlo algorithm is constructed to allow Bayesian inference in a hidden Markov model used in hematology. The algorithm has an outer Gibbsian structure, and incorporates both Metropolis and Hastings updates to move through the space of possible hidden states. While somewhat sophisticated, this algorithm still has problems getting around the infinite-dimensional space of hidden states because of strong correlations between some of the variables. A two-step variant of the Metropolis algorithm is introduced for posterior simulation.

Keywords: hidden Markov model, Metropolis algorithm, Gibbs sampler, Hastings algorithm, hematopoiesis

1. A MODEL

Suppose that each of N people in a room is holding a coin—the probability of heads for each coin is p . Independently of one another, each person flips his/her coin at random exponentially-distributed time intervals specified by a rate parameter λ . Over time, X , the number of facing heads, fluctuates between 0 and N . In fact, the state X is a continuous-time, finite-state Markov process whose stationary distribution is binomial(N, p). An observer, not knowing how many people are in the room, sees a sample of size n_1 with replacement from the faces of the N coins at time $t_1 = 0$. The observer records Y_1 , the number of observed heads, which given $X(t_1)$ has a binomial distribution with success probability $X(t_1)/N$. Similar observations Y_2, \dots, Y_m are made at subsequent time points $t_2, \dots, t_m = T$. Based on this time series of counts, the observer is asked to estimate the number of people in the room, the flip rate λ , and the success probability of the coins. Moreover, from a prior distribution on these three parameters, the observer is asked to derive the joint posterior distribution given the observed time series.

A model having precisely this probability structure has been proposed by Guttorp, Newton, and Abkowitz, 1990, (hereafter GNA), to describe time-series of proportions of a particular kind of cell in the bone marrow of a special kind of cat. In an experiment to study early cellular development in the blood-cell

system of a large animal, (see Abkowitz *et al.*, 1990), marrow samples are taken periodically (every 2 weeks or so) from female Safari cats. These cats are heterozygous for the X-linked enzyme glucose-6-phosphate dehydrogenase (G6PD), and because of X-chromosome inactivation early in embryogenesis, every cell in such a cat expresses either the paternal or maternal G6PD type. Thus every cell can be identified by its binary marker, the heads or tails from the coin model above. The proportion of paternal, say, cells in a sample of n_i cells is recorded at sampling time t_i . The sampled cells come from a special population of cells in the marrow; cells committed to a particular developmental lineage on the road to becoming mature blood cells. It is the evolution of this special population which is of interest to the investigator, but about which only limited information is available, through periodic sampling. In the model proposed in GNA, the population of interest is assumed to consist of a fixed number N of clones; a clone being all the descendants of a single cell. Some number X of these N clones are of the paternal type, because all the constituent cells have inherited the paternal G6PD type from their founding ancestor. The remainder are of the maternal type. Cell death and the activation of new clones force fluctuations in X , which is modeled like the fluctuating number of heads in the coin problem above.

In this paper, we construct a Markov chain for simulating a posterior distribution of the parameters N , p , and λ . A combination of Gibbs sampler and Hastings and Metropolis algorithms

form an algorithm similar to the hybrids of Tierney, 1991. Because the hidden state lives in an infinite-dimensional space, a straight Gibbs sampler is not possible. A fairly elegant hybrid algorithm is constructed, however it moves very slowly through the support of the posterior distribution. The essential problem is one that might have been foreseen: several of the variables are highly correlated *a posteriori*.

The *Metropolis-driven Hastings algorithm* is introduced as an alternative way to run a standard Metropolis algorithm for simulating posterior distributions.

First, we describe the deterministic likelihood calculations based on recursive updating techniques used by GNA.

2. RECURSIVE UPDATING

The likelihood function of $\theta = (N, \lambda, p)$ based on an observed series $y = (y_1, \dots, y_m)$ is

$$L(\theta; y) = P_\theta(Y_1 = y_1) \prod_{i=2}^m f_i(y_i | y_1, \dots, y_{i-1})$$

where

$$f_i(y_i | y_1, \dots, y_{i-1}) = P_\theta(Y_i = y_i | Y_1 = y_1, \dots, Y_{i-1} = y_{i-1}).$$

Marginally, the Y_i 's are not Markov, but this likelihood can still be simplified by noting that the i^{th} factor is a function of the $(i-1)^{th}$ factor. This is accomplished by expanding the i^{th} factor as a sum over the possible values of the hidden state $X(t_i)$ at that time point, and using the Markov properties of the hidden state (see GNA). A recursive algorithm, akin to a method in Baum, 1972, can be developed which takes advantage of this characterization. The algorithm requires calculation of transition probabilities of the hidden state between the sampling times. Defining $X_i := X(t_i)$,

$$P(X_i = k | X_{i-1} = j) = \sum_{l=0}^N \alpha_{jl} \alpha_{lk} \exp[-l\lambda(t_i - t_{i-1})]$$

where

$$\alpha_{ij} = (-1)^j (1-p)^{N/2} \sum_{\nu=0 \vee (j-1)}^{(N-i) \wedge j} \left(\frac{-p}{1-p} \right)^\nu a_\nu b_\nu,$$

and

$$a_\nu = \binom{i}{i-j+\nu}, \quad b_\nu = \binom{N-i}{N-i-\nu}.$$

When N is reasonably small ($N \leq 40$), calculation of the transition probabilities is quite feasible. Because of large negative and positive α_{ij} 's for large N , the calculation gets somewhat tricky and is subject to error. In GNA, likelihoods are computed on a grid of parameter values up to $N = 40$, and for different values of the other parameters. Without analytical results

about the shape of the likelihood surface, it is natural to search for other methods of computation. Markov chain simulation forms an important class of such methods, and a particular one is constructed to simulate the posterior of θ in this problem.

3. MARKOV CHAIN MONTE CARLO

The goal is develop an algorithm which simulates $\{\theta | Y\}$, the posterior distribution of the parameters θ given the data Y , and given a particular prior distribution, denoted $[\theta]$. (Following Gelfand and Smith, 1990, we use square brackets to denote the probability density, or more loosely the distribution, of the arguments.) Here, as is often the case, the Markov chain Monte Carlo method paradoxically involves increasing the dimension of the space in which computation takes place. The joint conditional distribution $[\theta, X_{[0,T]} | y]$ is actually simulated, where $X_{[0,T]}$ represents the unobserved Markov process; the number of heads in the room at all times during the sampling period $[0, T]$. The space is augmented so as to make each step in the algorithm as simple as possible. Following Tierney, 1991, or Geyer, 1991, an irreducible Markov process is constructed on the space of $(\theta, X_{[0,T]})$ such that the stationary distribution of this process is $[\theta, X_{[0,T]} | Y]$.

It is convenient to distinguish between the process X defined on the entire positive real line, and its restriction to the sampling window, $X_{[0,T]}$. The process X can be represented by the initial state $X_0 := X(0)$, a sequence of event times τ_1, τ_2, \dots , a sequence of deaths d_1, d_2, \dots , and a sequence of births b_1, b_2, \dots . The event times record times (in ascending order) at which a coin in the room gets flipped. The deaths and births determine changes that happen at the event times. The binary d_i indicates the face of the coin about to be tossed by the tosser at time τ_i . Thus $d_1 = 1$ indicates that the first coin tossed in the room was a head just before it was tossed. Similarly, the birth b_i indicates the outcome of the toss at time τ_i , 1 for heads and 0 for tails. Putting $\tau_0 := 0$, for any time t which happens to lie in $[\tau_i, \tau_{i+1})$,

$$\begin{aligned} X(t) &= X(\tau_i) & \text{for all } i \\ &= X(\tau_{i-1}) - d_i + b_i & \text{if } i \geq 1. \end{aligned} \quad (3.1)$$

If X_0 is binomial(N, p), then the X process is stationary, and so this is assumed throughout. Being minima of N independent exponentials with rate λ , the inter-event times $\tau_i - \tau_{i-1}$ are iid exponential random variables with rate $N\lambda$. Also, the births b_i are iid Bernoulli(p), and the deaths d_i have a Bernoulli distribution depending on the current state:

$$P[d_i = 1 | X(\tau_{i-1}) = x] = x/N.$$

This describes the state distribution $[X | \theta]$. The restriction $X_{[0,T]}$ of X to the sampling window $[0, T]$ is characterized by the first k events of X : k being the index of the largest event time smaller than T . Marginally, k has a Poisson distribution with mean $N\lambda T$.

Conditionally on the state, we have the *observation distribution*

$$[Y_i|X, \theta] = \text{Binomial} \left(n_i, \frac{X_i}{N} \right).$$

That is, the observations Y_i depend on the state X only through the state's values at the fixed sampling times. An approximate directed-graph representation of the model is shown in Figure 1.

The algorithm to simulate $[\theta, X_{[0,T]}|Y]$ has an outer Gibbsian structure in the that it proceeds by simulating $[\theta|X_{[0,T]}, Y]$ and then $[X_{[0,T]}|\theta, y]$.

4. THE PARAMETER UPDATE

Simulation of $[\theta|X_{[0,T]}, Y]$ is itself almost completely Gibbsian. The full conditional of the rate parameter is

$$[\lambda|N, p, X_{[0,T]}, Y] \propto e^{-N\lambda} \lambda^k [\lambda|N, p]$$

which is Gamma if the prior $[\lambda|N, p]$ is conjugate. The full conditional density of p given the rest, $[p|N, \lambda, X, y]$, is proportional to

$$p^{X_0 + \sum_{i=1}^k b_i} (1-p)^{N+k-X_0 - \sum_{i=1}^k b_i} [p|N, \lambda],$$

which is a Beta when the prior is conjugate. The parameter update would be completely Gibbsian if the conditional distribution of N given the rest was easy to simulate. In fact this is not the case, as $h(N) := [N|\lambda, p, X_{[0,T]}, Y]$ is proportional to

$$\begin{aligned} [N|\theta, p] & \left(\frac{N}{X_0} \right)^{N^k} ((1-p)e^{-\lambda})^N \\ & \times \prod_{i=1}^m \left(\frac{X_i}{N} \right)^{y_i} \left(\frac{N-X_i}{N} \right)^{n_i-y_i} \\ & \times \prod_{i=1}^k \left(\frac{X(\tau_{i-1})}{N} \right)^{d_i} \left(\frac{N-X(\tau_{i-1})}{N} \right)^{1-d_i} \end{aligned}$$

for N no less than all $X(\tau_i)$. A Hastings update of N is possible by generating a candidate N^* from a driving density $q(N, N^*)$ and then computing

$$r = \frac{h(N^*) q(N^*, N)}{h(N) q(N, N^*)}$$

and moving to N^* with probability $r \wedge 1$ (the minimum of r and 1). A simple q is Poisson, centered at the current value, and truncated at 0, which has

$$q(m, n) = \frac{m^n e^{-m}}{n!(1 - e^{-m})}.$$

Therefore given the hidden state and the data, parameter update is a three-step procedure. First λ and then p are sampled from

their full conditionals. Finally, a candidate update N^* is sampled from

$$N^* \sim \text{Poisson}(N) 1[N^* > 0]$$

and either N^* is accepted or N is kept, depending on a test using the ratio r above.

5. THE STATE UPDATE

Because the number k of events within the sampling window is random, the state $X_{[0,T]}$ actually lives in an infinite dimensional space. This space represents all the possible coin-flipping histories of the room over the sampling period. Gibbs sampling cannot work for such a point process, but other Markov chain simulation schemes can be devised. We describe three chains, which, if used in isolation, are not irreducible. When used together, they produce a combined chain which is irreducible and which has $[X_{[0,T]}|\theta, Y]$ as its stationary distribution.

5.1. The addition/deletion chain

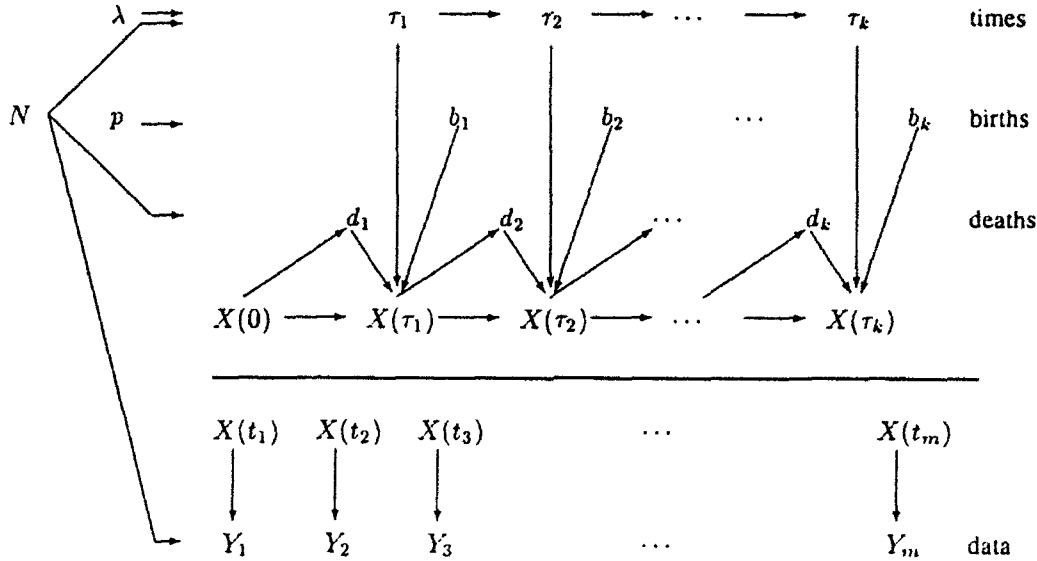
The first chain allows changes in the number of events in the sampling window $[0, T]$. In particular, it involves changing the number of *tied* events—the set of which is denoted

$$I_{\text{tied}} = \{i : d_i = b_i \text{ and } \tau_i \leq T\}.$$

These events are important because the state X remains constant across a tied event time. Suppose first that I_{tied} is not empty, so $k > 0$ and at least one of these events is a tied event, and introduce a user-supplied probability $a \in (0, 1/2)$. The transition kernel Q_1 determining the probability distribution of the first chain comes from the following rule. Add a tied event with probability a , and drop a tied event with probability a , and do nothing with probability $1 - 2a$. If an event is to be added, introduce a death d^* and a birth b^* which are tied, ($d^* = b^*$), that take place at an event time τ^* sampled uniformly in $[0, T]$. The common value of $d^* = b^*$ is chosen at random from the possibilities given the current state. For instance, if $\tau_i < \tau^* < \tau_{i+1}$ and $X(\tau_i) = N$, then $d^* = b^* = 1$ with Q_1 -probability 1. There are no tails in the room to be killed, only heads. This describes the addition of a tied event. If a tied event is to be dropped, then select its index i at random from I_{tied} and simply remove that event from the record. In the case of no tied events, change the selection probabilities. That is, add with probability a and do nothing with probability $1 - a$.

The addition/deletion algorithm always takes you from the current state X to a *candidate* state X^* which is a possible coin-flipping history of the room. This algorithm does not produce an irreducible chain, because although the number of events in $[0, T]$ can fluctuate arbitrarily, the level of the state can never change. With respect to a particular dominating measure, the

Figure 1: A directed-graph representation of the model: For instance, the births b_i all have parent p , are conditionally iid given p , and together with the deaths determine the level of the state. A death d_i has a distribution determined by the current state, and in turn affects the probability distribution of the future state.



density q_1 of the transition kernel Q_1 has the form

$$q_1(X, X^*) = \begin{cases} a/(J+1) & \text{if deletion,} \\ a/2 & \text{if non-boundary addition,} \\ a & \text{if boundary addition,} \\ 1-2a & \text{if } X^* = X \text{ and } J > 0, \\ 1-a & \text{if } X^* = X \text{ and } J = 0, \\ 0 & \text{otherwise.} \end{cases}$$

Here, J is the cardinality of I_{tied} for X , and the conditions on the right say how X^* came from X in the addition/deletion chain. A boundary addition is one where the level of X equals 0 or N at the event time τ^* . The density at a non-boundary addition is half that at a boundary addition because there are two possible values of $d^* = b^*$ in that case, as opposed to one for a boundary addition. Note that q_1 is not symmetric, so this algorithm is necessarily Hastings rather than Metropolis.

Upon generating a candidate X^* from the addition/deletion chain, we compute

$$r = \frac{f(X^*) q_1(X^*, X)}{f(X) q_1(X, X^*)}$$

where $f(X) = [X|\theta, Y]$ is the conditional density of the state X . The importance of adding and deleting ties is now apparent because the observation distribution is the same for both X and X^* , (the level of X is not changed in X^*). Thus

$$r_1 := \frac{f(X^*)}{f(X)} = \frac{[X^*|\theta]}{[X|\theta]}.$$

These marginal probabilities can be computed from the considerations of section 3. Noting that the deaths and births are independent of the event times,

$$[X|\theta] = p(X_0) \prod_{i=1}^{\infty} p(\tau_i|\tau_{i-1}) \prod_{j=1}^{\infty} p(d_j|X(\tau_{j-1}))p(b_j).$$

In forming the odds ratio r_1 , all factors associated to event times $\tau_i > T$ cancel because no changes occur in the state outside the sampling window. In fact, we need only conceptualize the existence of such an infinite sequence, as it never needs to be used in the computations. If X^* comes from an addition to X at a time when the level is x , we have

$$r_1 = N\lambda(px/N)^{b^*}((1-p)(1-x/N))^{1-b^*}.$$

Similarly, if X^* comes from a deletion in X of event i having level $X(\tau_i) = x$, then

$$r_1 = [N\lambda(px/N)^{b_i}((1-p)(1-x/N))^{1-b_i}]^{-1}.$$

Combining r_1 with the definition of q_1 , one complete step of the addition/deletion algorithm involves generating X^* from Q_1 and then computing r from above. With probability $r \wedge 1$, move to X^* , else stay put.

5.2. The level-change chain

A second chain is derived by changing the outcome of coin tosses in a certain way, while keeping the number k of coin

tosses in $[0, T]$ fixed. The transition kernel of this second *driver* is denoted Q_2 . This section describes the construction of a Q_2 which is symmetric. In this implementation, Q_2 is not used as a simple driver of a Metropolis algorithm, rather it is used as the basis of what we call a Metropolis-driven Hastings algorithm which is described in section 6.

The basic idea behind Q_2 is to change births and deaths locally in time. It is important that the changes are local because if a single change at one event time modifies the entire future level of X , then the computation of the odds ratio involves all the data beyond that event time. By making local changes in deaths and births that only affect the level of X in a small time interval, the odds ratio computations stay very simple. Of course, if you make a change by simply applying a binary switch to a single birth or a single death, then this can indeed change the entire future level of X . The trick is to make switches at randomly selected event times, then to counteract these switches by making complementary changes in the births and deaths at the next event time. This means that level changes are restricted to times t in between two event times; they are local rather than global changes in level.

More specifically, the level-changing chain works as follows. An index i is sampled at random from the set $\{0, 1, \dots, k\}$. First consider the case $0 < i < k$ which is slightly different from the boundary cases. Select two deaths d_i^*, d_{i+1}^* and two births b_i^*, b_{i+1}^* at random from all the legitimate possibilities given $X(\tau_{i-1})$ and $X(\tau_{i+1})$. In the candidate X^* , the level at t in $[\tau_i, \tau_{i+1})$ is, from equation 3.1,

$$X^*(t) = X^*(\tau_i) = X(\tau_{i-1}) - d_i^* + b_i^*.$$

The first death and the first birth must keep $X^*(\tau_i)$ within one step of both $X(\tau_{i-1})$ and $X(\tau_{i+1})$ and of course in between 0 and N . Also d_i^* must be consistent with $X(\tau_{i-1})$. Further, the second death and birth must satisfy

$$d_{i+1}^* - b_{i+1}^* = X^*(\tau_i) - X(\tau_{i+1})$$

and the death must be consistent with $X^*(\tau_i)$. All these various constraints define the legitimate possibilities of death/birth pairs given X . The resulting X^* differs in level from X only in $[\tau_i, \tau_{i+1})$.

The boundary case $i = k > 0$ just involves the first equation above. That is, a single death d_k^* and a single birth b_k^* are sampled at random from the legitimate possibilities so that for t in $[\tau_k, T]$,

$$X^*(t) = X^*(\tau_k) = X(\tau_{k-1}) - d_k^* + b_k^*.$$

When $i = 0$ and $k > 0$, we are updating X_0 , and in fact are dealing only with the second equation from above. Get d_1^* and b_1^* which keep the future level of X^* the same as the future level of X . That is, choose d_1^* and b_1^* at random from all legitimate possibilities and put

$$X_0^* = X(\tau_1) + d_1^* - b_1^*.$$

The final case is $i = k = 0$. In this case, there are no births or deaths in X during the sampling window, and so the update X^* comes from X by changing X_0 by 0, 1, or -1 .

The Q_2 driver takes a state X into another state X^* which is a possible coin-flipping history of the room. As with Q_1 , this Markov driver is not irreducible. This time because although arbitrary level shifts are possible, the number of events k in the sampling window always stays fixed. In contrast to Q_1 , Q_2 does have a symmetric density q_2 . To see why, consider the case where X^* comes from X through Q_2 and the modified index i is not on the boundary (an argument for the boundary case is analogous). The output X^* can be different from X only in the configuration of deaths and births at two subsequent event times. All event times and all other deaths and births must be the same for X and X^* . In fact, there is a finite set S of all states which equal X except possibly in the configuration of the deaths and births at the two event times of interest. This set S includes both X and X^* . By the random choice of legitimate possibilities in Q_2 , all states in S are equally likely outcomes of the level-change chain when any state in S is used as input. Hence

$$q_2(X, X^*) = q_2(X^*, X) = 1/\text{card}(S).$$

The use of Q_2 as a driver to simulate $[X|\theta, Y]$ is described in section 6. Before seeing this, we study one more method for transforming the state X .

5.3. The time-shifting chain

The final method for moving through the state space of X keeps fixed both the number k of events in $[0, T]$ and all the births and deaths at those k event times. This final driver Q_3 changes only the location in time of a randomly selected event, keeping it within $[0, T]$. Specifically, an index i is sampled at random from $\{1, 2, \dots, k\}$. If $i < k$, then a new state X^* is derived by sampling a τ_i^* uniformly from the interval (τ_{i-1}, τ_{i+1}) . If $i = k$, then the time τ_i^* is sampled uniformly from the interval (τ_{k-1}, T) . The new state X^* is identical to the current state X except for the i^{th} event time. If $k = 0$, then Q_3 is the identity map.

Symmetry of the density q_3 of Q_3 follows from an argument analogous to the one of the previous section. Let S be the (infinite) set of all states which are identical to X except possibly in the location of the i^{th} event time. All states in S are equally likely outputs from any input state in S ; thus symmetry. The driver Q_3 is used as input into a Metropolis update. The odds ratio depends only on the observation distribution, since any X and X^* pair have the same density $[Y|\theta]$. This ratio is

$$\begin{aligned} r &= \frac{[Y|X^*, N]}{[Y|X, N]} \\ &= \prod_{j \in I} \left(\frac{X^*(t_j)}{X(t_j)} \right)^{y_j} \left(\frac{N - X^*(t_j)}{N - X(t_j)} \right)^{n_j - y_j} \end{aligned}$$

where I_i is the index set of all sampling times t_j that are affected by the movement of τ_i to τ_i^* :

$$I_i = \{j : t_j \text{ is in between } \tau_i \text{ and } \tau_i^*\}.$$

In combination, Q_1 and Q_2 can reach any state from any other state in the space of $X_{[0,T]}$ given enough steps. The third chain Q_3 is added to speed up movement through this space, but is not necessary for the global chain to be irreducible.

6. A METROPOLIS-DRIVEN HASTINGS ALGORITHM

Suppose that a symmetric driver Q is available for sampling candidate states X^* from the state space. A Metropolis algorithm to simulate a chain with stationary distribution $[X|\theta, Y]$ involves sampling X^* from $Q(X, \cdot)$ and moving to X^* with probability $r \wedge 1$ where

$$r = \frac{[X^*|\theta, Y]}{[X|\theta, Y]}.$$

The Metropolis algorithm creates a Markov chain induced by the driving chain with kernel Q and using the single rejection test above.

An alternative, two-step algorithm is possible in any model where the Metropolis algorithm can be used to simulate a posterior. As above, use Q to generate a candidate state X^* , and then run a Metropolis algorithm on the marginal distribution of X . That is compute

$$r_1 = \frac{[X^*|\theta]}{[X|\theta]}$$

Let $Z = X^*$ with probability $r_1 \wedge 1$, and X otherwise. This marginal Metropolis algorithm has its own kernel H , and Z was generated from $H(X, \cdot)$. Although Q has a symmetric density, H does not. However, we can still use H as the driving chain for a Hastings algorithm whose stationary distribution is the conditional distribution of interest. Take Z and compute

$$r_2 = \frac{[Z|\theta, Y] h(Z, X)}{[X|\theta, Y] h(X, Z)}.$$

With probability $r_2 \wedge 1$, move to Z else stay at X . In fact r_2 is quite easy to compute, as it simplifies nicely to depend only on the observation distribution:

$$r_2 = \frac{[Y|\theta, Z]}{[Y|\theta, X]}.$$

This *Metropolis-driven Hastings algorithm* is different from a standard Metropolis update because it breaks the test into two parts; one depending on the marginal distribution of the state, and the other depending on the observation distribution. Since the first stage is a chain on the prior distribution, the method appears to be like sampling from the prior to simulate a posterior, however the prior sampling is directed rather than random. The relative merits of this two-stage method over the standard method have yet to be explored.

7. IMPLEMENTATION AND PROBLEMS

Computer code has been developed to implement the algorithm described in the previous sections, however it was not fully operational at the time of writing. Some general properties have been determined—most notably that the algorithm appears to be very slow, especially for large N . In running thousands of complete scans, each one including every updating chain, very strong serial dependence was observed on many of the component variables. The cause of this dependence is the shape of the posterior distribution over the state space. Basically, an attenuated *cigar-shaped* distribution exists. The rate parameter λ is highly associated with the number of events k . Because we use an outer Gibbsian structure, the overall chain has trouble moving both k and λ simultaneously.

To see how this problem arises, ignore conditioning on the data (in reality we do not want to do this since this is the algorithm's *raison d'être*). The problem stated above is equivalent to the problem in the simpler model described below. Consider the joint distribution of two variables λ and k defined by a marginal and conditional distribution

$$\begin{aligned}\lambda &\sim \text{Exponential}(\text{mean} = 10) \\ k|\lambda &\sim \text{Poisson}(\text{mean} = N\lambda)\end{aligned}$$

where N is constant. If a Gibbs sampler is used to simulate this joint distribution, the full conditional distributions must be sampled. One of these is Poisson, given above, and the other, the distribution of λ given k , is Gamma. Contours of this joint distribution are shown in Figures 2 and 3 for two different values of N . As N increases, we see a strong attenuation of the joint distribution into a *cigar shape*. It is well-known that Gibbs samplers have difficulties with such distributions, because updates sample from conditional distributions parallel to the coordinate axes, and thus never move very far. A standard trick is to reparameterize the model so that the coordinate axes form the major and minor axes of the dominant ellipsoidal shape. There are two reasons why this does not work here. Firstly, one of these variables, k , is an integer. Secondly, and perhaps more importantly, in the main model of this paper, the number of events k is just a summary of an entire state X . To reparameterize k and λ , we would have to somehow modify the entire state, and λ simultaneously. It would appear that the only way to do that is to abandon the outer Gibbsian structure altogether.

ACKNOWLEDGEMENTS

Conversations with Charlie Geyer and Chris Ritter have enhanced our understanding of Markov chain simulation methods.

REFERENCES

- [1] Abkowitz, J. L., Linenberger, M., Newton, M. A., Shelton, G. H., Ott, R. L., and Guttorp, P. (1990). Evidence

Figure 2: Marginal density of k and λ when $N = 5$: contours are at -3, and -5 units of log density from the mode, and the vertical dashed line represents the median of the marginal distribution of k .

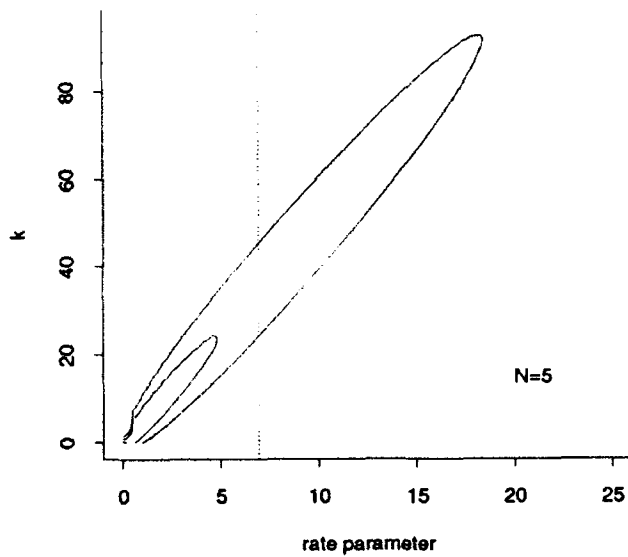
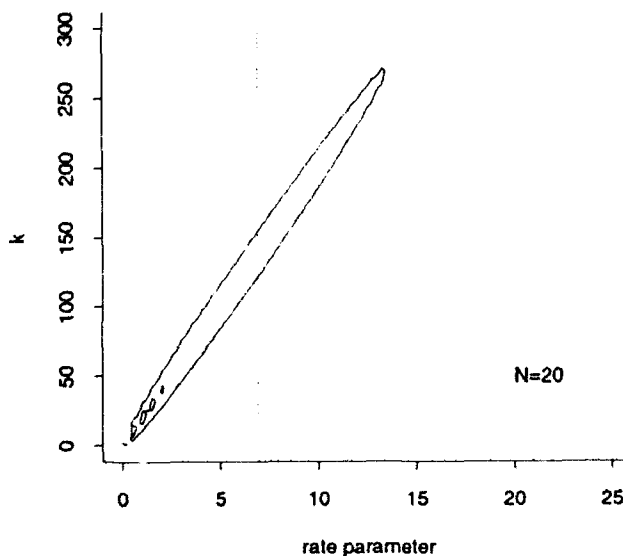


Figure 3: Same as Figure 2, but with $N = 20$.



for maintenance of hematopoiesis in a large animal by the sequential activation of stem-cell clones, *Proc. Nat. Acad. Sci. USA*, **87**, 9062-9066.

- [2] Baum, L. E. (1972). An inequality and associated maximization technique in statistical estimation for probabilistic functions of Markov processes. *Inequalities* **3**, 1-8.
- [3] Gelfand, A., and Smith, A. F. M. (1990). Sampling based approaches to computing marginal densities, *J. Amer. Statist. Assoc.*, **85**, 398-409.
- [4] Geman, S., and Geman, D. (1984). Stochastic Relaxation, Gibbs distributions and the Bayesian restoration of images. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, **6**, 721-741.
- [5] Geyer, C. (1991). Markov Chain Monte Carlo Maximum Likelihood, *Proceedings of the 23rd Symposium on the Interface*, 156-163.
- [6] Guttorp, P., Newton, M. A., Abkowitz, J. L. (1990). A Stochastic Model for Hematopoiesis in Cats, *IMA Journal of Mathematics Applied in Medicine and Biology*, **7**, 125-143.
- [7] Tierney, L. (1991). Markov Chains for Exploring Posterior Distributions. *Technical Report No. 560*, School of Statistics, University of Minnesota.

Data manipulation in perl

Douglas M. Bates

Department of Statistics

University of Wisconsin – Madison

E-mail: bates@stat.wisc.edu *

ABSTRACT

We are fortunate to have many different tools – statistical packages, spreadsheets, or database systems – for analysis and presentation of data. But often the first step in an analysis is manipulating or massaging the original data into a form that can be read by the package. Experienced Unix^a users may use *sed*, *awk*, or shell scripts for this. Recently a more powerful language, *perl*, has been made freely available. It provides a superset of the capabilities of *sed*, *awk*, and *sh* as a scripting language. We describe its use for simple data manipulation and for querying a membership directory. Finally we outline a more complicated query program in *perl* for the *Current Index to Statistics*.

Keywords: *sed*; *awk*; shell scripts; database query; Current Index

^a Unix is a registered trademark of Unix System Laboratories

1. INTRODUCTION

Often the first step in a data analysis consists of manipulating the raw data into a form that can be read by a statistical package or database system. For a single analysis on a small data set, this manipulation can be carried out with a text editor but editing such files by hand becomes tedious and error-prone when the data set is large or when there are many data sets to process. It then becomes worthwhile to create a special purpose program or script to perform the manipulation. On Unix systems this will often be done with shell scripts that use the stream editor, *sed*, or the *awk* programming language (Aho, Kernighan and Weinberger, 1988).

Recently Larry Wall of Jet Propulsion Laboratories created a language called *perl* — the Practical Extraction and Report Language. This powerful language for creating scripts provides the facilities of *sed*, *awk*, and the standard shells. It simplifies writing data manipulation scripts because you only have to learn one language instead of three or four. It also provides capabilities not in any of these other languages.

In section 2, we give a brief outline of the *perl* language and follow that with a couple of simple examples in sections 3. and 4. In section 5, we describe some of the advantages of *perl* and conclude with discussion of two larger applications that are more in the line of tools to enhance statistics research than statistical applications. These are query programs for the Joint Statistical Directory and the *Current Index to Statistics* databases.

2. A BRIEF OUTLINE OF PERL

The source code for *perl* is freely available via anonymous ftp on the Internet from sites such as *prep.ai.mit.edu* (in the directory *pub/gnu*) or *ftp.uu.net* (in the directory *pub/languages/perl*). It can be straightforwardly installed on most workstations. The source file contains the source for an extensive manual entry (about 90 pages long) that describes the syntax and use of the language. A more complete reference is the book "Programming perl" (Wall and Schwartz, 1990).

The simplest uses of *perl* involve reading one or more text files a line at a time, changing the line in some fashion, and sending the result to an output file. Usually, the *perl* program is stored in a file but simple, "one-liner" applications can be written on the command line after a *-e* flag as in *sed* and *awk*.

Three types of variables are used in *perl*: scalars, whose names must begin with the *\$* character; arrays, whose names must begin with *@*; and associative arrays whose names must begin with *%*. As in *awk*, a scalar can be either a numeric value or a character string, depending on context. There are many special variables in *perl*. The most important is *\$_* which is the default argument for many functions. Its value is usually the contents of the current line.

Algorithms in *perl* are expressed with functions and control structures. The syntax is very rich so we will introduce specific functions and control structures as we need them.

A *perl* program to process one or more text files usually has the form

```
#!/usr/bin/perl
# initialize any variables
```

* This research supported by the National Science Foundation under research grant DMS-9005904.

```
while (<>) {
    chop;
    # process the line
    # print the result
}
# final processing if needed
```

As in shell scripts, the `#` character introduces comments. The first line here is a special comment that allows the name of the script to be used as a command. Most shells for Unix systems adopt the convention that a file with the execute bit set (with `chmod +x`) and beginning with the characters `#!` is treated as input for the program specified immediately afterwards. One set of options can also be specified on this line. We assume that the perl compiler is stored as `/usr/bin/perl` or linked to that name.

The while loop is testing for the availability of another line to process. Generally, reading a line from a file handle in perl is indicated with broken brackets as in

```
$this_line = <STDIN>;
```

When no file handle is specified, the convention is to check for arguments specified on the command line, treat them as names of files, and try to open them for reading. If there are no arguments (actually if the array `@ARGV` is empty the first time a line is requested), the file handle `STDIN` is used instead. This reproduces the behaviour of many Unix tools that accept input from files whose names are given as arguments and otherwise accept input from the standard input stream.

To make it easier to write simple scripts, the flag `-p` to perl indicates that it should behave as in the loop above with the current value of `$_` being printed at the end of the loop. The flag `-n` has a similar meaning except that `$_` is not printed; the programmer must explicitly call a print function to produce any output.

When perl reads a line from a file, it retains the newline character at the end of the line. Since we often want to remove that character, it is common to "chop" the line immediately after reading it. The function `chop` removes the last character in a string. If no argument is given to `chop`, it removes the last character in `$_`.

3. CHANGING FIELD DELIMITERS

As a simple example, suppose you have a data file with entries like

```
5,3,001,000,692
5,2,138,000,698
5,3,146,000,698
...
```

where each line represents a single case and the values of different variables for that case are separated by commas. This is a typical organization of data as a *flat-file* where the *field delimiter* is a comma. You may find, though, that the package you are

using requires the fields to be separated by blanks instead of commas. In other words, you want the file to look like

```
5 3 001 000 692
5 2 138 000 698
5 3 146 000 698
...
```

Here we develop a simple tool for changing the field delimiter in such flat files.

3.1. Commas to blanks

A person with experience using `sed` would make the change from commas to blanks with

```
sed -e 's/,/ /g' < comma.dat > blank.dat
```

The edit command, given after the `-e` flag, causes a global substitution of commas by blanks. A perl equivalent is almost identical

```
perl -p -e 's/,/ /g;' < comma.dat > blank.dat
```

If there are several files, say `in1.dat`, `in2.dat`, ... to be manipulated in this way, the `sed` script can be used as

```
sed -e 's/,/ /g' in*.dat > blank.dat
```

but this would direct the output from all the input files to a single file. Often we want to take each of the input files and create a separate output file for it, something we cannot do directly with `sed`. But perl does allow such "in-place" editing with a `-i` flag

```
perl -i -p -e 's/,/ /g;' in*.dat
```

Each file will be overwritten with its edited version. If this seems a little too risky to you, you can change the `-i` flag to

```
perl -i.bak -p -e 's/,/ /g;' in*.dat
```

Now the files will be edited in place but the original will be preserved under a name created by adding the extension `.bak` to the current name. This allows you to back out of unanticipated and undesired changes.

As an example of such an undesired change, you may discover that the file can contain comments as well as data and you want to preserve the comment lines intact. If the comments are introduced by a `#` character as the first character on the line, it is easy to cause those lines to be preserved by adding an unless clause to the substitute command as in

```
perl -i.bak -pe 's/,/ /g unless /^#/' in*.dat
```

The pattern following the unless keyword is a regular expression, similar to those used in other Unix tools, that only matches lines with a `#` character at the beginning of the line.

3.2. White space to commas

The preceding example is rather trivial; we would expect any kind of editing tool to be able to replace commas by blanks. But consider the opposite operation of taking a file whose fields are delimited by white space and creating a comma-delimited file.

It may not be sufficient to use a simple text substitution where any occurrence of a blank is replaced by a comma since there may be multiple spaces between fields or a tab character may be used instead of a space or there may be spurious spaces or tabs at the end of a line.

Here we can use some facilities of `perl` that are more like the `awk` language which splits each input line into *fields*. The simplest version of the `split` function in `perl` splits the input line into fields delimited by white space. The inverse to `split` is `join` so it is tempting to think that

```
perl -i.bak -p -e 'join(',', split);' in*.dat
```

would accomplish the desired transformation. But it turns out that this command results in the output file being a copy of the input file.

To understand why this command does not cause any changes, we have to remember that the `-p` flag means to print the current contents of `$_` at the end of the loop. This was useful to us in the earlier examples because a substitution like `s/,/ /g` is applied to `$_` in place. But `join` is a function that creates a new character string. Unless it explicitly printed or assigned to a variable, it will be discarded. We can still make the delimiter transformation a one-liner by doing the printing explicitly and changing the `-p` flag to `-n`. The command is then

```
perl -ne 'print join(',', split), "\n";' in*.dat
```

The newline character must be added explicitly since the `perl` `print` function does not implicitly add one. We indicate the newline character by `"\n"`. There is a subtle difference between strings enclosed by single quotes and those enclosed by double quotes. Substitution of patterns like `\n`, `\t`, ... or variable names by their corresponding special characters or string values is only done on strings enclosed by double quotes.

3.3. A general delimiter transformer

By this point things are getting a little complicated for a one line command. Also, we are getting close to a general method for changing delimiters — `split` on the current delimiter and `join` with the new delimiter. To exploit the generality, we should create a script, say `delim.pl`, and explicitly perform operations such as cycling over the input lines. Such a script is

```
#!/usr/bin/perl -i.bak
$in = ' ';
$out = ',';
while (<>) {
    chop;
    print join($out, split(/$in/)), "\n";
}
```

The `perl` code defines two character strings, `$in` and `$out`, and uses them in the `split` and `join` functions. A sample usage would now be

```
delim.pl in*.dat
```

Because we now use variables to define the input and output delimiters, we can change the behaviour of the script. One of the most convenient ways to do this is on the command line itself with flags like `-i` for the input delimiter, `-o` for the output delimiter, and `-c` for the pattern indicating a comment line that should be preserved intact. Since the command line is available within the `perl` script as the array `@ARGV`, we could write code to check for the various flags and their arguments but it is much simpler to use the standard `perl` library subroutine `Getopts`. This subroutine takes a string specifying which single character options are allowed and whether they take a value. It signals the presence of an option by assigning a value to the variable `$opt_i` for the flag `-i`. It also removes the flags and their arguments, if any, from the array `@ARGV`. Using this subroutine, the script becomes

```
#!/usr/bin/perl -i.bak
require 'getopts.pl';
&Getopts('i:o:c:');
$in = $opt_i ? $opt_i : ' ';
$out = $opt_o ? $opt_o : ',';
while (<>) {
    if ($opt_c && /$opt_c/) {print; next;}
    chop;
    print join($out, split(/$in/)), "\n";
}
```

As in the C programming language (Kernighan and Richie, 1988), the line

```
$in = $opt_i ? $opt_i : ' ';
```

is equivalent to

```
if ($opt_i) {$in = $opt_i;} else {$in = ' '};
```

In checking for the comment pattern we first check to see if `$opt_c` is defined. If we didn't do this and no pattern was given, we would be matching against the null pattern and that would always be a successful match. The program would preserve every line intact.

A sample usage to change commas to blanks but skipping lines starting with `#` would be

```
delim.pl -i ',' -o ' ' -c '^#' in*.dat
```

4. FIXED FORMAT FIELDS

The few lines of data shown in the previous section were obtained from an administrative data file. When I receive this file it is in the form

```
Smith, John A.      LS  5   3  001   000   692
Jones, Mary E.      LS  5   3  001   000   692
Thompson, J. Walter LS  5   3  001   000   692
Miller, Susan       LS  5   3  001   000   692
...
```


where the person's name occupies the first 20 characters of the line, the college occupies the next 4, the level occupies the next 2, and so on. This is an example of a fixed format data file.

This file is convenient for a person to read but not as convenient for a statistical package to read. If we wish to read such data into a package like S (Becker, Chambers and Wilks, 1988; Chambers and Hastie, 1991) we have to indicate that the name should be treated as a single unit by enclosing it in quotation marks. We want to transform the file to look like

```
"Smith, John A." "LS" 5 3 001 000 692
"Jones, Mary E." "LS" 5 3 001 000 692
"Thompson, J. Walter" "LS" 5 3 001 000 692
"Miller, Susan" "LS" 5 3 001 000 692
...
```

Because the surname and the given names are separated by blanks and because everyone does not have the same number of given names, we cannot approach this by splitting the input line into fields delimited by white space. Instead we use the `unpack` function to separate the fields according to position. Here is the program that transforms the first data file into the second.

```
#!/usr/bin/perl -i.bak
$format = "A20 A4 A2 A5 A5 A6 A6";
while (<>) {
    undef @rec;
    foreach (unpack($format, $_)) {
        s/^\s+//;
        if (/^\d+$/) {push(@rec, $_);}
        else {push(@rec, '"' . $_ . '"');}
    }
    print join(' ', @rec), "\n";
}
```

The `$format` variable specifies the format of the line. Here an "A" followed by a field width indicates that the field is to be interpreted as an ASCII character string with trailing white space suppressed. An array, `@rec`, will be used to accumulate the fields before printing. Since values will be added to it by pushing them onto the end, it must be initialized to a null array at the beginning of the loop. The input record is then split with the `unpack` function producing an array value. The `foreach` control structure cycles through the elements of the array assigning each value in turn to the variable `$_`. Assignment within the `foreach` loop does not change the value of `$_` outside the loop.

The substitution line `s/^\s+//` strips leading white space from the current field. If the field consists solely of digits, it is pushed onto the `@rec` array as it is, otherwise it is surrounded by quotation marks before being pushed onto the array. Finally the array is joined into a single string and printed.

Here the format of the line is hard-coded into the program. This program could be made more general by using the subroutine `Getopts` to pick up the format from the command line. Two other enhancements are to make the pattern for a numeric

value more general and to force a conversion to a numeric type before pushing the value onto the array. Here is such a modified program

```
#!/usr/bin/perl -i.bak
require 'getopts.pl';
&Getopts('f:');
die "A format string must be specified.\n"
    unless $opt_f;
while (<>) {
    undef @rec;
    foreach (unpack($opt_f, $_)) {
        s/^\s+//;
        if (/^([+-]?[d+\.]?[d*]?\.\.d+)$/)
            {push(@rec, $_ + 0);}
        else {push(@rec, '"' . $_ . '"');}
    }
    print join(' ', @rec), "\n";
}
```

that produces the output

```
"Smith, John A." "LS" 5 3 1 0 692
"Jones, Mary E." "LS" 5 3 1 0 692
"Thompson, J. Walter" "LS" 5 3 1 0 692
"Miller, Susan" "LS" 5 3 1 0 692
...
```

As you might imagine, it is not easy to formulate the pattern that determines whether a string looks like a number. Even the one given above is incomplete because it does not take into account cases where an exponent is given. A alternative approach is to recall that the only reason for the double quotes is to protect embedded white space in text fields. We can change the whole algorithm to check for the presence of white space rather than something that looks like a number. Here is the alternate version

```
#!/usr/bin/perl -i.bak
require 'getopts.pl';
&Getopts('f:');
die "A format string must be specified.\n"
    unless $opt_f;
while (<>) {
    undef @rec;
    foreach (unpack($opt_f, $_)) {
        s/^\s+//;
        $_ = '"' . $_ . '"' if /\s/;
        push(@rec, $_);
    }
    print join(' ', @rec), "\n";
}
```

5. REASONS TO USE PERL

The examples I have given have only begun to illustrate some of the uses of `perl`. Some of the reasons that you may want to

use **perl** as a data manipulation language are:

Perl is freely available. Larry Wall has chosen to make **perl** freely available under the same conditions that the Free Software Foundation applies to their software. The code is well maintained and supported. There is the documentation for the language mentioned in section 2, and there is an active Usenet newsgroup `comp.lang.perl` discussing the evolution of the language and programming methods in the language.

The free availability of the source code means that you can install new releases as soon as they become available and you can install it on any machine that you wish. Considering the complexity of the language, it is remarkably easy to install on a workstation. There are also precompiled versions of **perl** for VMS, for MS-DOS and for the Amiga.

Libraries and a symbolic debugger. Many library subroutines are included with **perl**. The use of `Getopts` has been illustrated in section 3, and the `look` subroutine will be illustrated in section 6. The language has been designed to facilitate the use of libraries. The `require` and `provide` functions make it easy to load libraries. In addition, the conventions for symbol names allow you to avoid name conflicts between symbols in a library with those in the main script.

The most important library is the `perl5db` library that provides a symbolic debugger for **perl**. Interestingly, the symbolic debugger is itself written in **perl**. One can use this to set breakpoints in a script, to examine or to change the values of variables, to step through the execution of a script, and so on. Once you have started to use such a tool, it is difficult to overestimate its value. For those who have gotten used to using a debugger like `gdb` or `dbx` from within GNU `emacs`, there is a `perl5db` `emacs` mode as well.

Tools for conversion of existing scripts. The standard **perl** distribution includes the programs `s2p` for converting `sed` scripts to **perl**, `a2p` for converting `awk` scripts, and `find2perl` for converting `find` scripts. These conversion routines will produce a **perl** script that produces the same output as the original script although it may not be the best **perl** code for doing so. They do provide a starting point for an idiomatic **perl** implementation, though.

The **perl** compiler and the code it produces are reasonably efficient. The scripts are as easy to write as shell scripts and tend to run faster than shell scripts because they run as a single process.

Access to many low-level facilities. Many of the functions in **perl** are patterned after functions in the C library. Within **perl** you have access to system administration files (`getpwent`, etc.), `dbm` files (accessed as associative arrays), information in directories (`opendir`, `seekdir`, `readdir`), and file status information (`stat`). You can also use sockets and shared memory calls on machines that support them. Sockets are used to

create client/server pairs that can run on different machines.

Reading and writing binary data types. If necessary, you can even read and write binary data types, either in the native byte order or in network byte order.

6. DATABASE SEARCHES

Within the last two years the American Statistical Association and the Institute for Mathematical Statistics have made two machine-readable databases available for purchase. The Current Index to Statistics database contains bibliographic information on books and articles of interest to statisticians. It presently contains information on more than 100,000 articles, books, or reviews from the last 12 years. It can be ordered from the IMS (send mail to `ims@ucdavis.edu`). The Joint Statistical Directory database contains membership information for the American Statistical Association, the Institute of Mathematical Statistics, and the Statistical Society of Canada. It can be ordered from the ASA.

6.1. Directory Lookups

The utility of both of these databases is enhanced simple query programs. Since it is much easier to write a query program for the directory, we consider that case first. The database is distributed as separate files for surnames beginning with each letter of the alphabet. Each line in the directory entry for one person and begins with "I " followed by the person's surname. Since a binary lookup on the name will be very quick in **perl**, all these files can be combined into a single file that we will call `AZ`. The file should then be sorted by

```
sort -f AZ > temp; mv temp AZ
```

because the entries in the original files are not strictly in the ASCII collating sequence. A simple query program would then look up an individual's entry by surname. Such a program is

```
#!/usr/bin/perl
require 'look.pl';
$DIR = "/usr/local/lib/ASA_members";
$Prompt = '=> ';
chdir($DIR) || die "Can't chdir to $DIR: $!\n";
open(DB, "AZ")
    || die "Can't open file ${DIR}/AZ: $!\n";
if (@ARGV) {for (@ARGV) {&do_query;}}
else {
    print $Prompt;
    while (<>) {
        chop;
        &do_query;
        print $Prompt;
    }
    print "\n";
}
```

```

sub do_query {
    $key = $_;
    &look(*DB, "| $_", 0, 1);
    while (($_ = <DB>) =~ /^\\| $key/i) {print;}
}

```

Considering what the program does, it looks remarkably compact. This is because most of the work is done by the library subroutine `look` which performs a binary lookup on the given file for the a key.

Going through the program in sequence, it declares the name of the directory that holds the database and the prompt to use. The working directory is then changed to the database directory. In this case it is not really necessary to do that but in other programs where more than one file from the database needs to be accessed, it is helpful to change the directory. Another `perl` idiom is illustrated here, functions such as `chdir` and `open` return 0 if they are unsuccessful. Only in those cases will the other operand of `||` be evaluated and that causes the program to halt with an error message. (The behaviour is similar to the `stop()` function in *S* (Becker et al., 1988).) It is common to read this as "change directory or die!".

This script can be used in two ways; the desired name or names can be listed on the command line or the script will go into a prompting loop. The `if` statement tests if any arguments have been listed on the command line. In either case, the subroutine `do_query` is called to look up the listing. The `look` subroutine positions the file at the first line which begins with a sequence that is lexically greater than or equal to the key. The characters "`|`" are prepended to the key because all records in the database start with those. Finally, all records that match the key are printed. For example,

```

=> watts, d
| WATTS, Donald G.      . . .
| WATTS, Donna Lucas   . . .
=>

```

There is much more information printed on each line but I have omitted that here in the interest of conserving space.

Some reasonable enhancements to this program are to add a special case of `q` to mean "quit" and the format the records as they are printed. This script and a version that formats the records a bit more pleasingly are available for anonymous ftp from the machine `wingra.stat.wisc.edu` (128.105.5.32) in the directory `pub/ASA`.

6.2. Current Index Lookups

Creating a query program for the Current Index database is considerably more complicated than a simple binary lookup in the membership directory. Since the scripts that Paul Tukey and I have created for the CIS database are rather long and complicated, I will not discuss them in detail but rather give the broad outlines.

To make it easier to distribute the database to customers on floppy disks, it is divided into two data files for each year. Typically these are about 0.5 Mb. in size. We combine the files for each year and leave them under a name like 80 for the data from 1980. Each line contains fields listing the citation source, the title, the authors' names, keywords, alternate spellings of authors' names, and alternate spellings of words in the title.

We first create an inverted index for all the non-trivial words (truncated to six characters maximum) that occur in the title, the authors' names, and the keywords. This is an intensive process that takes about an hour on a moderately fast workstation but it only has to be done once a year when a new version of the database is released. The index is a two-stage index. A master index lists all the keys and indicates which data files contain records matching that key. For each data file with that key, it lists the number of records matching the key and a pointer to the byte position in a secondary index that gives the byte positions of all of these records. At Ron Thisted's suggestion we adopted the convention that, when there is only one record in the file matching the key, the pointer in the master index is to the record itself, not to a secondary index. To save space in the index, all the data files are given single character abbreviations. A separate configuration file matches the abbreviations to the names of the data files and the index file.

An example may make things clearer. A sample record from the master index is

```
lindne C 2 163613 D 1 519124 J 1 1201041
```

This indicates that there are two citations in the file with abbreviation C, and one citation in both D and J. The "C" file is 79 and has a secondary index called 79.ix. The line

```
287293 421922
```

begins at position 163613 in this file. Those numbers are the byte positions of the beginnings of the two records with this key in the file 79. For files D and J a reference to a secondary index is not necessary because there is only one record to be indexed. The number 519124 gives to the byte position of the beginning of the record with this key in the file 80.

One advantage of this two level index method is that it keeps the master index to a manageable size. If every byte position for every file were listed in a single level index, it would be very difficult for a human to try to browse it to check on correctness. A more important aspect of this organization is that it makes it easier to intersect the set of references that match different keys. It is common for a user to want to find those citations that match all of a set of keys. For example, a favorite test case at Texas A&M is the following

```

> CIS
Keyword access to CIS database.
Type '?' for help.
=> time series timeslab
=====

```

```

Frequencies: time(4239) series(3104) timesl(4)
=====
[Number of combined matches in 87: 1]
=====
%AUTHOR   = H. Joseph Newton
%TITLE    = TIMESLAB: A time series analysis ...
%JOURNAL  = ASA Proc. of Busn. and Econ. ...
. . .

```

Note that there are more than 4000 citations for the key 'time', more than 3000 for 'series' and only 4 for 'timesl'. We want to find those citations which match all these keys. We can tell immediately that there are no citations containing 'timesl' before 87 so we don't even have to consider the individual citations with 'time' or 'series' before then. Similarly in the 87 file we only have to match the citations with 'time' or 'series' against the one citation with 'timesl'. Doing the intersections a data file at a time results in considerable time savings.

Once the index is established a server program can be started. As mentioned in section 5., perl scripts can bind to sockets and accept connections over sockets. This means that the access to the database can be divided between a client run by the user on one machine and a server daemon running perhaps on another machine. In this case, the server accepts a connection to a socket, forks a child process to handle the instance of the connection, and goes back to listening on the socket. The child process accepts requests of keys to lookup and returns the citations. All the interaction with the user of prompting, decoding the user's commands, and formatting the output is carried out by the client script possibly running on another machine.

The scripts for creating the inverted index, running the server and the client are available for anonymous ftp from wingra.stat.wisc.edu as the file pub/src/CIS.shar.Z.

It is remarkable that a language which can be used as readily as shown in the simple examples of section 3., can also perform such powerful feats as communicating through sockets or forking child processes. It means that learning perl to write simple data manipulation scripts is a good investment of time because you can continue to use the same language if your needs become more complicated.

REFERENCES

- Aho, A. V., Kernighan, B. W. and Weinberger, P. J. (1988). *The AWK Programming Language*, Addison-Wesley.
- Becker, R. A., Chambers, J. M. and Wilks, A. R. (1988). *The New S Language*, Wadsworth, Pacific Grove, California.
- Chambers, J. M. and Hastie, T. J. (eds) (1991). *Statistical Models in S*, Wadsworth, Pacific Grove, California.
- Kernighan, B. W. and Richie, D. M. (1988). *The C Programming Language (Second edition)*, Addison-Wesley.

Wall, L. and Schwartz, R. L. (1990). *Programming perl*, O'Reilly & Associates, Sebastopol, California.

DE — the Data Entry Program

Michael Conlon*

Department of Statistics, University of Florida, Gainesville, FL 32610 USA

Abstract

DE — the Data Entry program, was created to meet the needs of people involved in the collection and computerization of large quantities of forms-oriented material. Data entry software improves the management of data by preventing a large class of common errors which occur during the keying of data. DE has many features which make it useful for this work. DE runs on a variety of platforms — UNIX, VMS and PC. It has many field checking features — double entry, range checking, character set restriction, forced entry, table lookup, date and time fields — plus the usual auto entry functions — dup, increment, skip, and date stamp. Hierarchical data is supported through multiple record types. All DE commands are two keystrokes each and can be entered from a numeric keypad with one hand for sustained numeric entry. Mnemonic sequences are also available as is on-line help. All DE files and definitions are stored as simple ASCII files, which can either be fixed or delimited. DE can export its definitions as programs for other systems (SAS[1], perl[2]) to use in reading its files.

1 Introduction

DE is a data entry program for forms-oriented raw data. Forms-oriented data frequently occurs in health-related research where subjects are measured repeatedly. Such forms may be several pages in length, may be used several times for each subject (longitudinal data) and many forms may be used to report the data for a single subject in a study. Forms-oriented data is organized into fields, where each field contains the response for a particular question or measurement. Fields typically have associated definitions that indicate what are valid contents for the field. Values collected that do not fit these definitions are errors that must be corrected by human intervention during the data collection and management phases of any project. DE contains features for creating

field definitions, organizing fields into screens, screens into forms and forms into collections called projects.

DE was also written for heterogeneous computing environments. DE's predecessor GIF (General Input Facility, M. Conlon, 1986[3]) was used in clinical settings on laptop computers, as well as by professional data entry clerks operating in production data entry shops, but was limited to MSDOS computers. DE is a C language program which uses the *curses*[4] screen access library to achieve portability. *curses* is available in the public domain for a large variety of computing systems. DE has been compiled and tested under Ultrix, NeXT/Mach, MSDOS, VMS and AIX/370, and on a variety of keyboards. Little effort should be required to make DE run on other systems as well.

DE was also designed to be easy to learn and ergonomic to use. Previous experience with data entry software led to the development of a simple function key sequence paradigm that permits one-handed operation, and complete operation of the program from a numeric keypad. DE has on-line subject help and context sensitive help.

DE has the standard data entry software automatic functions such as auto-duplication of values from previous records, auto summing of fields, auto counting (sequencing), insertion of date and time stamps into records, as well as features to improve the management of data entry personnel. DE can keep log files of work done on various projects with time stamps and productivity measures.

In addition to standard data entry features, DE also writes programs in common statistical analysis languages that can be used to read the data files that DE produces. DE data files are always clear ASCII, but with many forms and variables it could be tedious to create SAS input statements for the DE files from a complex data entry system. DE solves this problem by writing complete SAS data step programs that can be used to create SAS datasets from DE data files. DE also writes access programs for perl. In this way, the records produced by data entry personnel can be read into statistical analysis and/or data management software without the

*Work on DE has been supported by NIDA grant R01-DA05854-01A2 and NIH grant R01-NR02444-01

typical delay involved in writing programs to read the data.

2 The Keyboard

The DE accepts input from the keyboard as characters to be entered into fields of on-screen forms. To signal a command to DE, the user presses one of four function keys, followed by a mnemonic single character. The four function keys are typically available in VT100 mode under UNIX (PF1, PF2, PF3, PF4), or as F1, F2, F3 and F4 on an MSDOS computer. The actual keys depend on the machine being used. When function keys are not available, for example NeXT, the functionality is obtained by using the top keys on the keypad as function keys through software written for this purpose.[5]

DE uses the names Help, Format, Record and Field for the four function keys to correspond to classes of functions. Help provides access to system-wide information and services. Format is used to access functions related with the collection of screen formats that make up the current project. Record functions are related to record-level operations, such as insert record, delete record, and search functions. The Field functions are directed at individual fields (erase, duplicate) and characters.

Functions are accessed by pressing a function key followed by a single alphabetic character. There is never any shifting involved, so one-handed operation is supported. By using key sequences predicted by function keys, all ASCII sequences are available for data entry. The alphabetic characters may be sent as upper or lower case, so the shift state of the keyboard never needs to be changed to type a function key sequence. Figure 1 shows the functions available under each function key.

The key sequences are easily learned and are listed on-line through the Help-K sequence. DE is exited by using Help-Q which automatically saves work. DE writes new records into the data file being created as soon as data entry on the record is finished. Explicit saving is never required. In this way work cannot be lost by power failure, system failure or operator negligence.

DE operates in one of four modes: Define, Enter, Examine and Verify. A mode corresponds to a task performed with DE. Define mode is used to define new screens. Enter mode is the default and is used for data entry. Examine mode is the same as Enter mode, except that changing the data is not allowed. Verify mode is a special mode for double entering data that is already on file. Verify mode has limited movement functions as entry is intended to proceed as done previously. A mode can be chosen from the command line when DE starts, or by mode switching function key sequences. Figure 2

shows the DE functions available in the DE modes.

3 Screens

Since DE uses *curses* to access the screen, screens of arbitrary size are automatically supported. On systems capable of creating them, DE can operate with screens of size larger than the MSDOS standard 80 columns by 24 rows. Thus screens can truly reflect the printed form and greatly improve the readability of the data entry screens. Snapshots of the screens can be taken and printed or written to files for comparisons with the hand-written paper forms. Screens of size 65x54 are useful for emulating the printed page since an 8.5" by 11" piece of paper has a 6.5" by 9" usable area after 1" margins are subtracted from four sides. At 10 characters to the inch and 6 lines to the vertical inch, this translates to 65 by 54 characters on the printed page. DE screens of this dimension thus imitate the printed page perfectly.

Creating DE screens for use in a data entry project is a three step process:

1. Screen images are created using an editor or DE define mode.
2. Field attributes are assigned using `dedef`.
3. Screens are tied together into a "project".

Each of these steps is described below.

DE screens can be created using any editor capable of producing ASCII files. It is done most easily with an editor that can also address the screen size being created. Any text can be put anywhere on the screen. DE reserves the first line on the screen for status information. This is shown in Figure 3. Data entry fields are marked on the screen using the underscore character (`_`). Consecutive underscores are considered to be single fields. Fields can be of any length. Once the ASCII image of the DE screen has been completed, it is run through `a2def`, a `perl` program supplied with DE. `a2def` creates a DE definition file from the screen image text. The definition file is used to assign attributes to each of the fields and is used by DE to redraw the screen for data entry. DEF files can also be created using DE define mode described in the DE manual[6].

Once a DEF file has been created, attributes can be added using `dedef`. `dedef` is a DE session that uses the DEF file as data and uses DE to allow the user to fill in "fields" that correspond to attributes of the fields in the screen being created. In this way DE operates on its own DEFs in a recursive manner. Assigning attributes to fields is always optional. Attributes can be added or

Help = PF1		Format = PF2		Record = PF3		Field = PF4	
Help	System Help	Help	Format Help	Help	Record Help	Help	Field Help
F	Field Info	S	Snapshot	D	Delete	C	Correct
C	Context	L	Single line	I	Insert	X	Delete Char
A	About DE	T	Top-of-form	R	Release	E	Erase
B	Basics	N	Jump to #	N	Jump to #	N	Jump to #
K	Keyboard	P	Project Info	F	Search	D	Dup
Q	Quit	R	Release	B	Search Back	S	Single corr
E	Enter Mode	W	Write File	↑	To prev rec	←	Home
V	Verify Mode			↓	To next rec	→	End
X	Examine Mode			←	To begin rec	↑	To prev fld
D	Define Mode			→	To end rec	↓	To next fld
T	Table info					A	Toggle Auto
						I	Toggle Insert

Figure 1: The DE keyboard

```

ENTER  REC   4 FMT   1 FLD   1 INS AUT first
Name : THIS IS A NEW FIELD_____
Addr1: _____
Addr2: _____
City : _____ ST -- Zip -----

```

Figure 3: A DE screen

changed at any time during the creation and use of the screen.

Screens are tied together into DE "projects" by creating a PRJ type file with an editor. A PRJ file lists the screens to be presented and can contain instructions regarding repetitions of the screens and how the records should be formatted in the data file. DE can produce delimited ASCII, where each value is separated from the next by a delimiter (often a TAB), or column-oriented ASCII in which each value occupies the same relative position on each record. Projects that consist of single screens — such as elementary questionnaires — do not require project files. In this case, a default project file is provided. Project files may contain any number of references to screens. Each reference is the name of the DEF file containing the screen and its field attributes. In this way, the same screen can appear in any number of projects. A sample project file is shown in Figure 4.

```

1,y
2,1,demographics,demo,1,2
2,2,visit,visit,0,3
2,3,exit,exit,1,1

```

Figure 4: A DE project file

The type 1 record is a header that controls file formats. In this case the *y* indicates that a fixed column file is to be created. The type 2 records define the screens. Three screens are referenced. The first is stored in a file named *demographics.def*. Records created using this DEF will be marked with a record identifier at the front of the record containing the text "demo". The screen is to be used once and followed by the second screen. The repeat count on the *visit* screen of zero indicates that any number of visit records can be entered. The operator will have to release the format to proceed to the exit form.

4 Files

All files used and created by DE are clear ASCII. This makes them easy to transport between various systems and makes them directly usable by other software such as editors. DE uses a variety of file types to contain information. A list of the types is provided in Figure 5.

DAT and LOG files are created by DE during normal use. DEF, PRJ and TBL files are created prior to use to define the data entry work to be done. TBL files can be created using *detbl*, which like *dedef* uses DE to create

Function	Keys	Mode			
		Enter	Examine	Verify	Define
About Help	Help-Help	•	•	•	•
Field info	Help-F	•	•	•	•
Context help	Help-C	•	•	•	•
Table Info	Help-T	•	•	•	•
About DE	Help-A	•	•	•	•
Keyboard Help	Help-K	•	•	•	•
Goto Define	Help-D	•	•	•	•
Goto Verify	Help-V	•	•	•	•
Goto Examine	Help-X	•	•	•	•
Goto Enter	Help-E	•	•	•	•
Quit	Help-Q	•	•	•	•
Format help	Fmt-Help	•	•	•	•
Snapshot	Fmt-S	•	•	•	•
Single line	Fmt-L	•	•	•	•
Top of form	Fmt-T	•	•	•	•
Jump to format	Fmt-N	•	•	•	•
Format release	Fmt-R	•	•	•	•
Write file	Fmt-R	•	•	•	•
Project info	Fmt-P	•	•	•	•
Help record	Rec-Help	•	•	•	•
Next record	Rec-↑	•	•	•	•
Prev record	Rec-↓	•	•	•	•
Search forward	Rec-F	•	•	•	•
Record release	Rec-R	•	•	•	•
Search backward	Rec-B	•	•	•	•
Insert record	Rec-I	•	•	•	•
Delete record	Rec-D	•	•	•	•
Jump to record	Rec-N	•	•	•	•
Home record	Rec-←	•	•	•	•
End record	Rec-→	•	•	•	•
Help Field	Fld-Help	•	•	•	•
Field Correct	Fld-C	•	•	•	•
Single correct	Fld-S	•	•	•	•
Jump to Field	Fld-N	•	•	•	•
Dup Field	Fld-D	•	•	•	•
Erase field	Fld-E	•	•	•	•
Toggle Auto	Fld-A	•	•	•	•
Toggle Insert	Fld-I	•	•	•	•
Home field	Fld-←	•	•	•	•
End field	Fld-→	•	•	•	•
Next field	Fld-↓	•	•	•	•
Prev field	Fld-↑	•	•	•	•
Delete char	Fld-X	•	•	•	•
Insert char	any	•	•	•	•

Figure 2: DE functions and Modes

File type	Usage
.DAT	Data. The records entered by the operator are stored here.
.DEF	Definitions. Field definitions are stored here. One line is used per field. One DEF file is created for each screen in the project.
.PRJ	Project file. This file controls the order of record formats and the format of the data file.
.TBL	Table file. Used to store tables of information for field validation.
.VFY	Verify file. Contains the record number of the last successfully verified record in the data.
.LOG	Log file. Contains a record of the use of the corresponding project.

Figure 5: DE file types

```

1, Five point Likert Item
2, 1, Strongly Disagree
2, 2, Disagree
2, 3, No Opinion
2, 4, Agree
2, 5, Strongly Agree

```

Figure 6: A DE Table file

table definitions for attachment to fields. Tables are lists of appropriate values for fields with optional descriptors on the values. A typical table file for a 5 point Likert type item is shown in Figure 6.

The first entry on each line is a record type. A table file has two types of records, a table identifier record and a collection of table entry records. The table entry records each have record type 2, followed by a value, followed by a descriptor. If this table is stored in a file called `likert5.tbl` and `likert5` is entered as a table name for a field during DEF file creation, then when the operator is using DE for data entry, only values appearing in this table will be allowed in the field. An attempt to enter a value not in the table will result in an error message. The operator can view the table from DE during data entry to be reminded of the valid field values. Note that "blank" is not a legal value in this

table. This implies that a value for the field is required. If no value is permitted, a mechanism must be provided for the operator to enter the "no value." If "no value" is permissible, a table entry must be created that defines the permissible "no value" value.

The VFY file is automatically maintained by DE during verify mode. Verify mode is used to perform double-entry verification of data that has already been entered once. During double-entry, the operator is presented with a blank form, and as characters are typed, DE checks them against the values previously entered and on file. If the characters agree, verification proceeds. When a conflict occurs, DE signals an error with a message on the screen, and the operator must override the value typed or the value on file. As records are completed in verification the VFY file is updated to contain the record number of the highest record number verified. In this way verification can resume at the appropriate spot in the file should verification be interrupted.

5 Availability

DE is distributed under a GNU[7] general license via anonymous ftp from `banana.stat.ufl.edu` as `de.tar.Z`. C language source, makefiles, examples, supporting perl scripts and manual (TeX and PostScript) are included in the distribution.

References

- [1] SAS Institute, Inc. *SAS Language and Procedures: Usage, Version 6, First Edition*, 1990.
- [2] Larry Wall and Randal L. Schwartz. *Programming perl*. O'Reilly & Associates, Inc., 1991.
- [3] Michael Conlon. *GIF: General Input Facility*. Department of Statistics, 1986.
- [4] C.R. Hoare. *Screen Updating and Cursor Movement Optimization: A Library Package*. UC Berkeley, 1976.
- [5] Brian Burton. *TCKEYS*. Penn State University, 1991. computer files.
- [6] Michael Conlon. *DE: The Data Entry Program*. Department of Statistics, 1992.
- [7] Richard Stallman. *Emacs reference manual, 5th edition*. Free Software Foundation, 1986.

Writing Unix Commands

Phil Spector
Department of Statistics
University of California, Berkeley
Berkeley, CA 94720

Abstract

One of the most attractive features of the UNIX operating system is its extensibility, that is, the ability to add new or modified commands to the collection of commands which is already part of the operating system. This paper will discuss a variety of ways to write commands to run in the UNIX operating system, beginning with collecting a set of commands in a file to execute in a "batch" mode, passing arguments to a set of commands in a file and concluding with an outline of how to pass arguments to a C program through the command line in the fashion of standard UNIX commands.

1. What is a UNIX command?

Any file on a UNIX file system can be invoked as a command, provided that it has been marked as executable. To make a file executable, issue the UNIX command "chmod +x filename". If an executable file contains text, then it is interpreted as a series of shell commands, that is, a set of commands which will be understandable by the command interpreter known as the shell. Alternatively, commands can be created by compiling a program written in a programming language such as C or Fortran.

2. Conventions for UNIX Commands

Most UNIX commands are invoked by a command name, followed by zero or more options, which are specified as a minus sign followed by a single letter. Some options require additional arguments; others are flags which signal a particular feature of the program should be invoked. Filenames are often not preceded by a minus sign/letter combination, are usually the last argument on the command line. Figure 1 illustrates some examples.

3. Shell Scripts

As mentioned previously, an executable file containing text is assumed to contain commands which will be interpretable by the shell. Specifically, by default, it is

Figure 1: Examples of UNIX Commands

<code>cc pgm.c</code>	creates executable (a.out)
<code>cc -c pgm.c</code>	creates object file (pgm.o)
<code>cc -o pgm pgm.c</code>	creates executable name pgm

assumed that the commands will be interpretable by the Bourne shell (/bin/sh). To override this default, the first line in your shell script should be of the form:

```
#!/program_name
```

as in

```
#!/bin/csh
```

to specify that the statements contained in the file should be interpreted by the C shell (/bin/csh). The shell scripts in this paper are all assumed to be interpreted by the Bourne shell.

The simplest shell scripts just consist of a series of commands identical to those which would be typed in to a terminal in a usual interactive session. For example, we could create a shell script containing the lines:

```
cc -o prog program.c -lnag
prog < prog.in > prog.out
vi prog.out
```

to allow us to compile, run and view the output from a program with a single command. Even such simple scripts can be useful, for example, when several commands need to be run in a specified order – the shell script can be run in the background, and the order in which the commands are run is guaranteed by the order they are entered in the script. However in most cases, it is useful to pass information through command line arguments. When writing a shell script, you have access to the command line arguments through the following scheme. The symbol \$0 represents the name of the command which invoked the script; the symbols \$1, \$2, . . . , \$9 represent the first nine arguments to the script, and \$* represents all the arguments. As always when dealing with a shell, special care should be taken when using special characters such as *, \$, [,] and ". If you don't want these characters to have their special meaning, they should be preceded by a backslash (\

), or enclosed in single quotes (''). As a simple example of a shell script with arguments, consider a program to print a specified field from a file, where a field is defined as text in the file separated from other text by any number of white spaces (blanks or tabs). The UNIX program `awk` can do this very simply; in `awk`, fields are simply referred to as `$1`, `$2`, etc.; the conflict between the use of the dollar sign in the shell and in `awk` can be resolved through careful use of the backslash. In this simple example, the field number is given as the first argument, and the filename as the second:

```
#!/bin/sh
# first argument is field number,
# second is filename
awk "{print \$${1}}" $2
```

One word of warning about names for shell scripts – avoid using the name `test`, since it is a built-in command in some shells, and it is often difficult to “convince” the shell that you’re referring to your command and not the shell’s.

4. Shell Services

To write effective shell scripts, it is important to be aware of the services which the shell provides. These features are automatically available when you execute your scripts, and make developing useful and flexible shell scripts very easy. These services include:

1. **Shell Variables** The shell allows you to create and manipulate arbitrarily named variables, either in response to the UNIX prompt, or within a shell script. To set a variable, use the form `variable=value`; there should be no spaces surrounding the equal sign. To refer to a variable in a script, precede it’s name with a dollar sign `$`. Sometimes it is important for other programs which are run to be aware of the values of shell variables. To make the value of a shell variable known to any shells which are created after the variable’s value has been set, you can use the command `export`.
2. **Wildcard Expansion** You can use a variety of special characters as part of filenames to represent any filename which matches a particular pattern. The simplest example of a special character used this way is the asterisk (*), which matches anything in a wildcard; for example `*.c` will be matched by any filename which ends with the extension “.c”. The important thing to remember about wildcard expansion is that it takes place when the command is interpreted, so by the time your shell script is reading the command line arguments, any wildcards

will have already been expanded into a list of file names which match the pattern given. To pass a literal special character to a shell script, precede it with a backslash (\), or surround it with single quotes ('').

3. **Redirection** To direct the output of your shell script to a file, instead of having it appear on the computer’s screen, you can use the greater than symbol (>) before the file name on the command line. Similarly, to read input from a file instead of the keyboard, you can use the less than symbol (<). Thus, when you write a shell script, you don’t have to worry about opening and closing files, since redirection can take care of most simple cases.
4. **Command Output** Inside a shell script, you can set a shell variable equal to the output from any UNIX command by enclosing the command in backquotes (`). If the output consists of more than one word, you can break them apart by using the `set` command as illustrated in this example:

```
$ set `date`
$ echo $1
Thu
$ echo $2
May
$ echo $3
7
...
```

Since this method overrides the previous meanings of the variables `$1`, `$2`, etc., you should be sure you have stored the command line arguments before using this technique inside a shell script.

5. **Testing Operators** The command `test` allows you to test relationships among shell variables. The form of the test command is “`test v1 op v2`”, where `op` is `=` for string comparisons and `-eq` for numeric comparisons. (Shell variables are basically strings, and special commands are necessary to make them be treated like numbers.) Expressions for test may also take the form “`test -key filename`” where `-key` can be `-f` to test if filename is a file, `-d` if it is a directory, etc. See the UNIX manual page for `sh(1)` or `test(1)` for more details.
6. **Shifting arguments** The shift command re-names the current arguments so that `$3` becomes `$2`, `$2` becomes `$1`, etc. This is useful for parsing arguments, as later examples will show.

7. **Echo command** The command `echo` writes text to standard output, which can be redirected if necessary to write to another location. By default, `echo` always terminates its output with a newline; this can be overridden with the `-n` option.
8. **Reading input** The command `read` reads a line from standard input and places it into a shell variable. The format is simply "read variable". This can be combined with the `echo` command to prompt a user for information:

```
echo -n "Name of file? "
read filename
```

The shell variable `$filename` will now be equal to the name of the file the user specified. Note the use of the `-n` option of `echo` to keep the prompt and the user's input on the same line.
9. **Control of Flow** The shell provides many flow control commands, such as `if-then-else`, `while`, `for`, etc. Some of these will be illustrated in sample shell scripts in the next sections.
10. **Arithmetic** While the shell is not very well suited for arithmetic, the `expr` command allows simple integer arithmetic. Keep in mind, however, that shell variables are essentially strings, even if some commands treat them like numbers. When using `expr`, spaces must be placed between operators and values, and the multiplication and division operators must be escaped with a backslash, that is, the symbol for multiplication is `*` and that for division is `\/`. The output of `expr` is especially well suited for trapping in backquotes, as will be shown in the second example.

5. A More Complex Example

Consider the field extraction example presented in Section 3.. Suppose we wish to enhance the command by allowing an optional field separator to be supplied (with the `-s` option), as well as lifting the restriction on the ordering of the arguments, by using the `-f` option to recognize the filename. The following shell script uses the features described above to achieve these goals, using the standard UNIX conventions mentioned in Section 2.:

```
#!/bin/sh
sep="" \
filename=""
f=0

for i in $* ; do
```

```
case $i in
-f ) f=$2
    shift
    shift ;;
-s ) sep="$2\"
    shift
    shift ;;
* ) filename=$1 ;;
esac
done

if test $f -eq 0
then
    echo "No field number given. Exiting ..."
    exit
fi

if test "$filename" = ""
then
    echo "No filename given. Exiting ..."
    exit
fi

awk "BEGIN{FS= $sep}{print \\\$f}" $filename
```

As is often the case with shell scripts and other user written commands, the bulk of the program processes arguments and traps errors, printing appropriate messages. As usual, backslashes are used to protect special symbols such as quotation marks and dollar signs. The `for` loop at the beginning of the file can provide a model for parsing command line arguments in the standard way.

6. Example Shell Script using Arithmetic

The following shell script writes the contents of one or more files to an output location, and writes an index of starting and ending lines to standard output. The name of the output file is read from standard input after a suitable prompt. The output of the `expr` command is used to perform the necessary arithmetic.

```
#!/bin/sh

start=0
end=0
file=""

echo -n 'Name of file for output: '
read out

rm -f $out

for i in $*
do
n=`cat $i | wc -l`
start=`expr $end + 1`
end=`expr $start + $n - 1`
cat $i >> $out
echo $i $start $end
done
```

The command `wc` is used to count the number of lines in each file before writing it to the output file. A possible enhancement to the script would be check for the existence of the output file before simply removing it.

7. Writing Commands in C

When the functionality of a command becomes too complicated for a shell script, or when maximizing efficiency becomes a critical issue, the C programming language can be used to write UNIX commands. Often a shell script can serve as a prototype for a command, allowing you to test it out and see if it is useful, before investing the time to develop a similar command in C. Since many standard UNIX commands are written in C, the language provides a wide range of features to support development of commands. Some of these features are discussed in the subsections below.

7.1. Command Line Arguments

To parse command line arguments in a C program, the main program should be declared with the two arguments `argc` and `argv` as follows:

```
main(int argc, char **argv)
```

When the program is invoked, `argc` will contain the number of command line arguments (including one for the command name itself), and `argv` will be an array of character pointers, each pointing to one of the command line arguments. (`argv[0]` is a pointer to the name of the invoked command itself.)

7.2. Conversion of Strings to Numbers

Since the command line arguments contained in `argv` are pointers to character strings, conversion is necessary before these arguments can be used as numbers. The most effective way to convert them is to use the C system routine `sscanf`, which allows a formatted read from a character string into any C object. To convert the first command line argument to a long integer, for example, we could use the following code:

```
long n;

sscanf(argv[1], "%ld", &n);
```

Note that the destination argument must be passed to `sscanf` as an address, since its value will, by necessity, be changed.

7.3. Opening Files

The basic routine for opening files is `fopen`. The following example illustrates its use:

```
#include <stdio.h>

FILE *fp;
char *filename, *type;

if((fp = fopen(filename, type)) == NULL){
    fprintf(stderr, "Can't open file %s\n",
        filename);
    exit(1);
}
```

The first argument is a pointer to a character string containing the name of the file you wish to open, and the second is a pointer to a character string describing the type of open: "r" for read only, "w" for write only, or "a" for append. Note that when the type is "w", an existing file of the same name will be overwritten. As implied by the example, `fopen` returns a NULL pointer if it can't successfully open the file. This possibility should always be checked when calling `fopen` and appropriate action taken if the open fails.

In addition to FILE pointers created through `fopen`, inclusion of the file `stdio.h` also gives you access to the three FILE pointers `stdin`, `stdout` and `stderr`, corresponding to standard input, standard output, and standard error, respectively.

7.4. Reading Files

Reading Character-by-Character To read a character at a time from a file represented by a FILE pointer, use the function `getc`. This function will return the

character which is read or the special value EOF (defined in `stdio.h`) when the end of the file is encountered. To read from standard input, you can either use `getc(stdin)` or function `getchar()`, which is equivalent.

Formatted Reads The function `fscanf` allows formatted reading from a file. It accepts a variable number of arguments: the first is a `FILE` pointer, followed by a character string containing formatting information. The remaining arguments are the addresses of the objects which are to receive the values described in the formatting string. Some of the codes which can appear in the formatting strings are shown in Table 1.

Table 1: Codes for Formatting Input

Code	Type	Code	Type
%d	integer	%s	character string
%ld	long integer	%f	float
%h	short integer	%lf	double
%c	single character		

In each case, the format code is enclosed in quotation marks, and the final arguments to `fscanf` must be the *address* of the intended target. For example, to read two integers into the variables `m` and `n`, and then to fill an array with the `m`×`n` values which follow the integers, the following code could be used (assuming that `fp` had been set to an appropriate value by `fopen`)

```
long i,j,n,m;
double x[100][100];
FILE *fp;

fscanf(fp,"%ld %ld",&n,&m);

for(i=0;i<n;i++)for(j=0;j<m;j++)
    fscanf(fp,"%lf",&x[i][j]);
```

Each item in the file must be separated by at least one occurrence of white space (blank, tab or newline). When reading from standard input, the function `scanf` can be used; its arguments are used in the same way as those of `fscanf`'s, but there is no file pointer argument.

Reading Line-by-Line If the items in your input file are not separated by white space, you may have to read an entire line of your file, and break it up as necessary. (A line is defined as a string of characters ending in a newline character.) To do this, you can use the function `fgets` (or `gets` for standard input). You must declare a character string long enough to hold the longest line of your file before calling `fgets`. The three arguments to `fgets` are the address of the buffer, the size of the buffer,

and the file pointer, as illustrated by the following code fragment:

```
#define BUFSIZE 1024
#include <stdio.h>

FILE *fp;
char buf[BUFSIZE]

while(fgets(buf,BUFSIZE,fp) != EOF){
    ....
}
```

Both `gets` and `fgets` terminate the buffer with a null terminator, but `fgets` discards the newlines it encounters, while `gets` transfers them to the buffer.

8. An Example

As a simple example of a command written in C, consider a command to count the number of tabs (or some other character) in a file. Such a program would also be useful if written as a filter; that is, if no file name is specified, then the standard input (with `FILE` pointer `stdin` as defined in `stdio.h`) should be used as input. The `-c` option on the command line allows counting a character other than the tab. The following program shows one way to achieve this goal:

```
#include <stdio.h>
#include <ctype.h>

main(int argc,char **argv)
{
    FILE *fp;
    int i,n,nf,nfiles,argind[256];
    long tot;
    char chk = '\t';

    i = 1;
    nfiles = 0;
    while(i < argc)
        {if(argv[i][0] == '-')goto doopt;
         argind[nfiles++] = i;
         i++;
         continue;
        }
    doopt:
        if(argv[i][1] == 'c')
            chk = argv[i][2];
        else
            printf("%s unknown option.\n",argv[i]);
        i++;
    }

    tot = (long)0;
```

```

nf = 0;
do
{if(nfiles == 0)fp = stdin;
 else fp = fopen(argv[argind[nf]],"r");
 if(fp == NULL)
 {printf("can't open %s\n",
        argv[argind[nf++]]);
  continue; }

 n = 0;
 while((i = getc(fp)) != EOF)
   if(i == chk)n++;

 tot += (long)n;

 if(fp == stdin)printf("%8d\n",n);
 else printf("%8d  %s\n",n,
              argv[argind[nf]]);

 fclose(fp);
 nf++;
 }
 while(nf < nfiles);

 if(nfiles > 1)
   printf("%8d  total\n",tot);
}

```

The position of arguments representing filenames is stored in the array `argind`; since the array `argv` which is automatically allocated to hold the command line arguments, is not volatile, the filenames can be reliably found in that array for the duration of the program. The use of a `do-while` loop insures that the counting process will take place for at least one file. If no file is specified on the command line, the `FILE` pointer is set equal to `stdin`, allowing use of the command as a filter. Finally, when more than one file is processed, the command prints a total of the number of occurrences of the specified character in all the files considered.

9. References

- Kernighan, Brian W. and Pike, Rob (1984), *The UNIX Programming Environment*. Englewood Cliffs, NJ: Prentice-Hall, Inc.
- Kernighan, Brian W. and Ritchie, Dennis M. (1988), *The C Programming Language*, 2nd ed. Englewood Cliffs, NJ: Prentice-Hall, Inc.

Projection Pursuit Indices based on Fractal Dimension

Javier Cabrera and Dianne Cook

Dept of Statistics, Hill Cntr, Busch Campus, Rutgers University, New Brunswick, NJ 08904

cabrera@stat.rutgers.edu

Abstract

We propose two different projection pursuit indices that are derived from estimators of the fractal dimension. One index is based on the Grassberger-Proccaccia estimator of fractal dimension, and the other is derived through maximum likelihood estimation. These indices are designed to expose data lying on lower dimensional manifolds and other low level non-random structures. An example of such data comes from points generated by the RANDU random number generator which when put into a three-dimensional unit cube lie on 15 parallel planes [11].

1 Introduction

Projection pursuit is a data analytic tool for exploring multivariate data, which attempts to find interesting low dimensional projections. The original idea was proposed by Kruskal [12] but the phrase "projection pursuit" was coined in further work by Friedman and Tukey [3]. The procedure involves searching high (p) dimensional data for interesting low ($k = 1, 2$ maybe 3) dimensional projections through the optimization of a criterion function, called the projection pursuit index, which measures the interest in each projection.

These early notions of projection pursuit indices have been crystallized into the formulation of indices based on measures of deviation from normality [9],[7], [10]. Various indices have been proposed and implemented along these lines: Negative Entropy and Moment [9],[10], Fisher Information [8], L^2 -distances of the density of the projected data from a normal density [4], [6], [13], [2]. Each of these measures depends on some form of density estimation and variously uses kernel methods or expansions in terms of orthonormal functions. The kernel methods do well in 1-dimensional indices, because they can be computed rapidly, with the advantage of the Fast Fourier Transform, and they can find a large variety of structures by varying the bandwidth of the kernel. However for 2-dimensional indices the kernel methods are too slow computationally to be practical. On the other hand the orthonormal expansions methods are fast to compute for any dimension, with the proviso that the expansion is truncated within reason. Very low truncations pro-

duce indices which are adept at finding large structure such as clusters and outliers. Higher order truncations produce indices which are sensitive to fine structure, because they are, by construction, multimodal from the nature of higher order orthonormal functions. However this also makes them difficult to optimize [2].

Ideally, to complement the strengths of these indices, we'd like to construct an index which easily finds fine structure, in particular data which lies on a low dimensional (especially non-linear) manifold, and other low level non-random structure. To derive such indices we discard the notions of searching for deviations from normality and turn to ideas of data dimension in the literature on dynamical systems.

2 Fractal dimension

There are many quantities that go under the name "dimension." The one we will discuss here was introduced in Grassberger and Procaccia [5] for dynamical systems. For the purpose of this paper we are not going to give an account of dynamical systems, but we will introduce a few basic definitions.

Consider a set A generated by the sequence $\{P_i\}_1^\infty$ of points in the p -dimensional real space. Suppose that A is a subset of a manifold M . In the case of a dynamical system this is the orbit generated by iterating a smooth map from the manifold M into itself, starting at some initial point, P_1 .

Define dimension of A at a point P to be:

$$d(P) = \lim_{r \rightarrow 0} \frac{1}{\log r} \lim_{N \rightarrow \infty} \log \left\{ \frac{N_n(P, r)}{n} \right\},$$

where $N_n(P, r) = \#\{P_i: i \leq N, |P_i - P| \leq r\}$.

Notice that, even if manifestly invariant under smooth changes of coordinates, this dimension is not a geometric property of the set A but rather of the measure defined by the sequence $\{P_i\}_1^\infty$, in a neighbourhood of the point P .

The above definition implies that for large n and small R we can use the approximation:

$$N_n(P, r) \doteq C \cdot r^{d(P)}, \quad 0 \leq r \leq R,$$

where C is a constant which does not depend on r .

In practice, since we only observe a finite set of points, the calculation of the dimension is done by selecting a ball of radius R centered at P and measuring the slope of a log/log plot of the number of pairs of points at a distance less than r versus r , for a few values of $r \leq R$. This is:

$$\hat{d}_1(P) = \text{Slope}[\{\log(r_i), \log(N_n(P, r))\}_{i=1}^m], \\ 0 \leq r_1 < \dots < r_m \leq R$$

Clearly this calculation should be done only for a range of r 's which are sufficiently small to be close to the limit but which are large enough to contain a significant number of points. This estimator is similar to the one proposed by Grassberger and Procaccia [5].

The above approximation suggests that if we restrict the measure defined by the sequence $\{P_i\}_1^\infty$ to the ball with center at P and radius R , namely $B_R(P)$ we obtain the probability measure:

$$\text{Prob}[B_r(P)] = \lim_{n \rightarrow \infty} \frac{N_n(P, r)}{N_n(P, R)} = \left\{ \frac{r}{R} \right\}^{d(P)}$$

Then the function:

$$F(r) = \left\{ \frac{r}{R} \right\}^{d(P)} I_{[0, R)}(r) + I_{[R, \infty)}(r)$$

is a distribution function for the random variable r . We can obtain a simpler form for this function by applying the transformation $u = -\log(r/R)$. Then the variable u has an exponential distribution with parameter $d(P)$.

In practice we only observe a finite piece of the sequence, namely $\{P_i\}_1^n$, and only m of them $\{P_{(i)}\}_1^m$ are within a distance R of P . Their corresponding values of r are $\{r_i\}_1^m$, where $r_i = d(P_{(i)}, P) < R$, and let $u_i = -\log(r_i/R)$. Under certain assumptions about the original sequence we can treat the u_i 's as an independent sample and therefore we can estimate $d(P)$ by maximum likelihood:

$$\hat{d}_2(P) = \{\bar{u}\}^{-1} = \left\{ \sum_{i=1}^m \frac{\log(r_i)}{m} \right\}^{-1}$$

For more information in this topic and further explanation for the case of dynamical systems see Cabrera and Llave [1].

3 A new class of projection pursuit indices

Some of the underlying assumptions, such as ordered data, of the definition of dimension are not satisfied for multivariate data, so in a strict sense the dimension

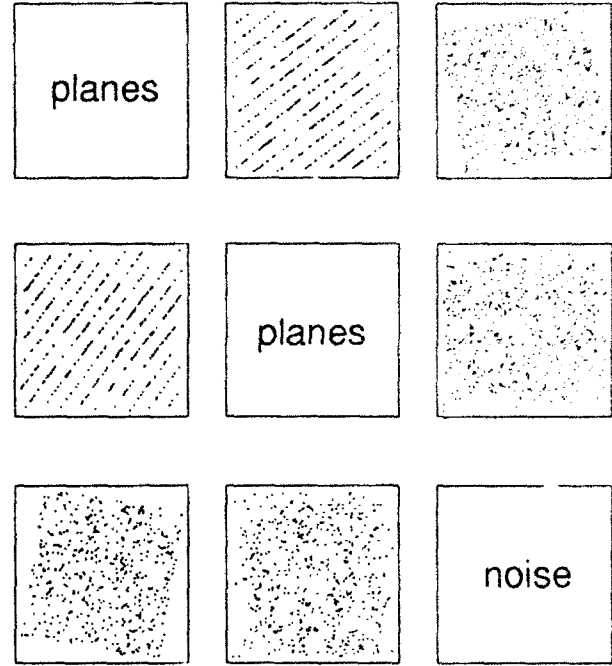


Figure 1: RANDU - matrix of pairwise plots

methodology does not apply. Fortunately the estimators do not rely on these assumptions, so we construct projection pursuit indices along these lines, and examine their performance in finding low level non-random structure in data. We propose two projection pursuit indices, derived from the two dimension estimators, each of which is calculated by taking their median value over the sample points. The median, rather than the mean, is used to ensure resistance to fluctuation in the values in boundary areas.

The first index, \hat{I}_1 , proposed is based on estimator $\hat{d}_1(P)$. To calculate $\hat{d}_1(P)$, at point P_i , the m nearest neighbours, $P_{(j)}$, are used to approximate the slope of $\log(j)$ versus $\log(r_j)$, $j = 1, \dots, m$, where $r_j = |P_{(j)} - P_i|$ is the Euclidean distance from P_i to $P_{(j)}$. This method of estimation requires that $R = \max\{r_j, j = 1, \dots, m\}$ is variable which we found to give better results than keeping R fixed. Also because the nearest neighbours are needed this method is heavy computationally.

The second index, \hat{I}_2 , is based on estimator $\hat{d}_2(P)$. In this case, R is fixed to a quantile q_α of the empirical distribution of pairwise euclidean distances between points. In our experiments this was sufficient to obtain good results without using computationally intense nearest neighbours schemes.

Finally, another way to speed up the computation of

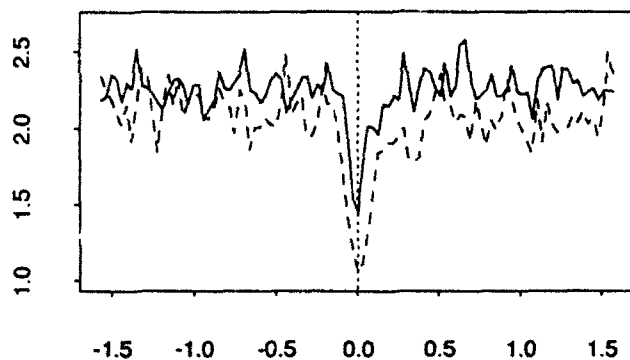


Figure 2: *RANDU* - solid: Index 1, dashed: Index 2

the indices is to compute the dimension estimators in a small (50 or 100) subset of the points only.

4 Examples

Although for most p -dimensional data it is not possible to visualize the entire 2-dimensional projection pursuit function, to a limited extent we can visualize 2-dimensional projection pursuit on 3 and 4-dimensional data.

We consider two examples to illustrate the behaviour of the fractal indices. The first is a sample from the infamous *RANDU* random number generator, used widely in the 1970's. It is based on the multiplicative congruential scheme: $x_{n+1} = (2^{16} + 3)x_n \pmod{2^{31}}$. Points generated by it lie on 15 parallel planes defined by $9x_n - 6x_{n+1} + x_{n+2} \equiv 0 \pmod{2^{31}}$, when sequentially placed into a 3-dimensional cube [11]. Figure 1 contains a matrix of pairwise plots of the sample. Figure 2 shows plots of the two indices, \hat{I}_1 (solid line) and \hat{I}_2 (dashed line), calculated over a 180° , $(-\pi/2, \pi/2)$, rotation of the data with the planar structure falling in the center, $\pi = 0$. Despite the jagged nature of the lines it is clear that the values fluctuate around a plateau of values close to two, and then drop dramatically near the planar projection. Both indices pick up this projection as being of dimension less than the rest. (The values for index 1 were calculated with $m = 20$, and index 2 was calculated with R fixed at the quantile generated by $\alpha = 0.05$).

Figure 4 contains contour plots and perspective plots of 2-dimensional projection pursuit with \hat{I}_1 , (a), (b), and \hat{I}_2 , (c), (d), on 4-dimensional data with points distributed on a spiral in the first two variables and random normal noise in the second two variables. This is pictured in a matrix of pairwise plots in Figure 3. The space of all 2-dimensional projections of \mathbb{R}^4 can be reparametrized in terms of two angles, (θ, ϕ) . Letting the angles each range from $(-\pi/2, \pi/2)$ gives the set of 2-projections $\{(\underline{\alpha}, \underline{\beta}) : \underline{\alpha}' = (\cos \theta, 0, \sin \theta, 0), \underline{\beta}' =$

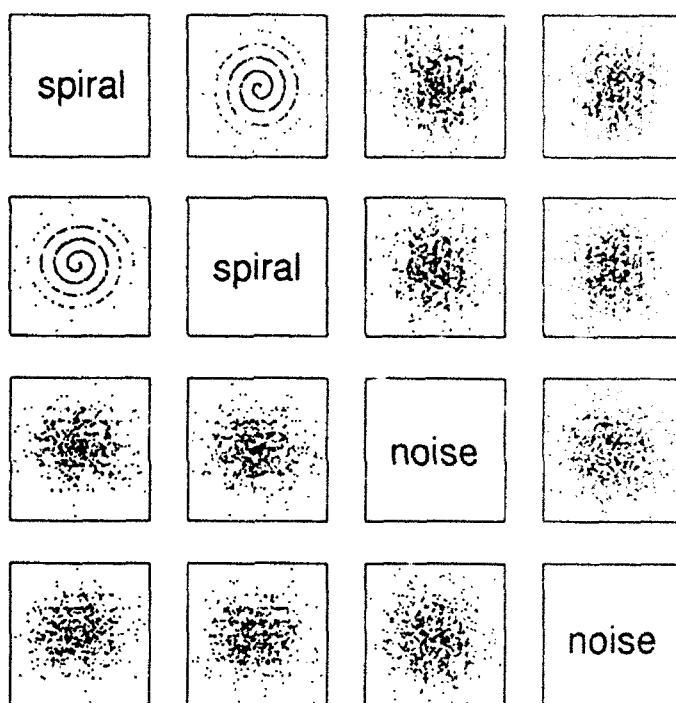


Figure 3: *Spiral* - matrix of pairwise plots

$(0, \cos \phi, 0, \sin \phi), \theta, \phi \in (-\pi/2, \pi/2)\}$. Then (θ, ϕ) form the axes of the plots in Figure 4 and the surface is generated by calculating the index over a grid of angle values. The surface, then, represents an interpolation between the first two variables and the second two variables. When $\theta = \phi = 0$ the corresponding projection is that of the first two variables, that is, the spiral. When $|\theta| = |\phi| = \pi/2$ the projection is of variables 3 and 4, whilst when $\theta = 0, |\phi| = \pi/2$ it corresponds to variables 1 and 4, and to variables 2 and 3 if $|\theta| = \pi/2, \phi = 0$. All the plots show a steep peak centered at $(0, 0)$ which means that both indices respond strongly to the projection containing the spiral. (The negative index values are plotted because the dip due to dimension reduction is best seen as a peak in the perspective plots. Index 1 was calculated with $m = 20$ and index 2 was calculated with R fixed at the quantile generated by $\alpha = 0.05$.)

In both examples we see that the second index, based on the maximum likelihood estimator of the fractal dimension, produces sharper changes at the projection containing the structure of interest. So this method is not only more computationally efficient but it also makes the structure easier to find.

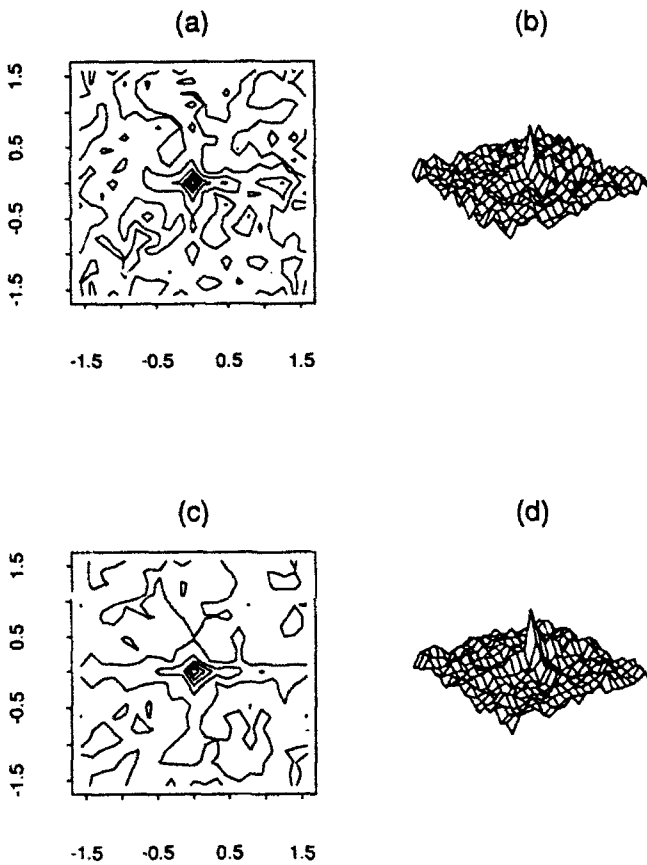


Figure 4: Spiral - (a), (b) Index 1; (c), (d) Index 2.

5 Discussion

We've seen that the indices clearly detect the low level structure of both the RANDU planes and the spiral in these two examples. So the idea of using dimension estimators has some merit in the development of projection pursuit indices with specific responsive to projections in which the data lies on or near low dimensional manifolds. There is still much to be refined about the indices that we have currently proposed.

Both estimators of fractal dimension fluctuate wildly between projections, perhaps due the utilization of the median and of the cutoff value, R . For various reasons, a chief one being ease of optimization, we would like to eliminate these fluctuations. Some of it was alleviated by using the same sample of points throughout the calculations. Other ways need to be explored before any practical implementations can be embarked upon.

All of the indices based on measures of deviation from normality require that the data be sphered before conducting projection pursuit. A major bonus of the fractal

indices is that they don't have this requirement, so solutions can be given in the original coordinate basis.

References

- [1] Cabrera, J., Llave, R. (1991) Attractor reconstruction methods for time series. Statistical elimination of noise. Submitted to *Statistical Computing*.
- [2] Cook, D., Buja, A., Cabrera, J. (1991) Direction and Motion Control in the Grand Tour. *Proc. of the 23rd Symp. on the Interface between Comput. Sci. and Statist.*, Springer-Verlag, New York.
- [3] Friedman, J. H. and Tukey, J. W. (1974) A Projection Pursuit Algorithm for Exploratory Data Analysis. *IEEE Trans. Comput. C* 23 881-889.
- [4] Friedman, J. H. (1987) Exploratory Projection Pursuit. *J. Amer. Statist. Assoc.* 82 249-266.
- [5] Grassberger P. and Procaccia I. (1984): *Dimensions and entropies of strange attractors from a fluctuating dynamics approach*, *Physica*, 13D, 34-35.
- [6] Hall, P. (1989) Polynomial Projection Pursuit. *Ann. Statist.* 17 589-605.
- [7] Huber, P. J. (1985) Projection Pursuit (with discussion). *Ann. Statist.* 13 435-525.
- [8] Jee, J. R. (1985) A Study of Projection Pursuit Methods. *Tech. Report TR 776-911-4-85*, Rice University.
- [9] Jones, M. C. (1983) The Projection Pursuit Algorithm for Exploratory Data Analysis. PhD Thesis, University of Bath.
- [10] Jones, M. C. and Sibson, R. (1987) What is projection pursuit? (with discussion) *J. Roy. Statist. Soc. Ser. A* 150 1-36.
- [11] Knuth, D. E. (1981) *The Art of Computer Programming. Vol. 2 Seminumerical Algorithms*. Addison-Wesley, Reading, Mass.
- [12] Kruskal, J. B. (1969) Toward a practical method which helps uncover the structure of a set of observations by finding the line transformation which optimizes a new "index of condensation". In *Statistical Computation*, (R. C. Milton and J. A. Nelder, eds), 427-440. New York: Academic Press.
- [13] Morton, S. C. (1989) Interpretable Projection Pursuit. *Tech. Report No. 106*, Lab. for Computat. Statist., Stanford Univ.

An Analysis of Polynomial-based Projection Pursuit

Dianne Cook^{††}, Andreas Buja[†], Javier Cabrera[†]

[†] Bellcore, 445 South St, Morristown, NJ 07962-1910

^{††} Dept of Statistics, Hill Cntr, Busch Campus, Rutgers University, New Brunswick, NJ 08904

dcook@stat.rutgers.edu

Abstract

Projection pursuit is a procedure for searching high dimensional data for "interesting" low dimensional projections via the optimization of a criterion function called the projection pursuit index. Two recently proposed indices, Legendre (Friedman, 1987) and Hermite (Hall, 1989), based on density estimation using expansions in terms of orthogonal polynomials, are particularly useful because they are computationally speedy. A general form for this type of index is introduced which highlights some problems and leads to a new index, the Natural Hermite. More practically we show that adjusting the truncation in the series expansion tailors the indices to be sensitive to different levels of structure. Low order indices have a "myopic" quality enabling them to find large structure from a distance, whilst higher order indices are fine tuners, able to find intricate structure in data but needing to be much closer to see it.

1 Projection Pursuit

Our interest in the structure sensitivity of projection pursuit indices stems from implementing projection pursuit dynamically (Cook, Buja, Cabrera, 1991) in XGobi, which is dynamic graphics software being developed by Swayne, Cook and Buja (1990). The projection pursuit algorithm can be watched as it searches for interesting projections. Through watching this procedure, using different indices and parameters on a variety of data, we became curious about differences in the types of structure uncovered. Before elaborating on these differences we need to briefly describe the projection pursuit procedure and development of the indices.

The term "projection pursuit" was coined by Friedman and Tukey (1974) to describe a procedure for searching high (p) dimensional data for "interesting" low ($k = 1$ or 2 usually, maybe 3) dimensional projections. The procedure, originally suggested by Kruskal (1969), involves defining a criterion function, or index, which measures the "interestingness" of each k -dimensional projection of p -dimensional data. This criterion function is searched over the space of all k -dimensional projections of p -space, for global and local maxima. The result-

ing solutions ideally illuminate low dimensional structure in the data not necessarily found by methods such as principal component analysis.

"Interesting" structure is equated with non-normality by Jones (1983) and Huber (1985). They propose projection pursuit indices which are measures of deviation from a null model of normality. Integral to this model is the notion of affine invariance of the indices. This suggests discarding structure such as location, scale and covariance, which can be found reasonably well with other methods, by sphering the data before beginning projection pursuit. Consequently we have a framework for considering a family of projection pursuit indices based on an assumption that a p -dimensional data vector Z is a random variable on \mathbb{R}^p , such that $EZ = 0$ and $\text{Cov}Z = I_p$.

From Z we want to construct a k -dimensional projection pursuit index. For simplicity, let $k = 1$, so we consider all 1-dimensional projections of Z ,

$$Z \longrightarrow X = \alpha'Z \in \mathbb{R} \quad (\alpha \in S^{p-1}),$$

where S^{p-1} is a unit $(p-1)$ -sphere in \mathbb{R}^p . Then X is a random variable on \mathbb{R} . In the null case if $Z \stackrel{d}{\sim} \mathcal{N}(0, I_p)$ then $X \stackrel{d}{\sim} \mathcal{N}(0, 1)$. Let the random variable X have distribution function $F(x)$ and density $f(x)$. Then an index, I , is constructed by measuring the deviation of $f(x)$ from a standard normal density.

A useful, practical index of this kind was proposed by Friedman (1987), but he detoured from the above route by first mapping X into a bounded interval $[-1, 1]$ by the transformation $Y = 2\Phi(X) - 1$, where Φ is the distribution function of a standard normal. By doing this he hoped to concentrate attention on differences in the center producing an index robust to tail fluctuations. In the null case if $X \stackrel{d}{\sim} \mathcal{N}(0, 1)$ then $Y \stackrel{d}{\sim} U[-1, 1]$. Let Y have distribution function $G(y)$ and density $g(y)$. Friedman's proposed index is an L^2 -distance of $g(y)$ from the density of $U[-1, 1]$:

$$I^L = \int_{-1}^1 (g(y) - \frac{1}{2})^2 dy \quad (11)$$

We call this the Legendre index because $g(y)$ is expanded

in terms of the natural polynomial basis with respect to $\mathcal{U}[-1, 1]$, Legendre polynomials.

Friedman's approach suggests a framework for a general class of indices, I , based on L^2 -distances between the data density, $f(x)$, and a null density, $\phi(x)$. There are two important details of this class of indices:

1. I is a functional of f .
2. f depends on the projection vector, $\underline{\alpha}$, so that projection pursuit entails the search for local maxima of I over all possible $\underline{\alpha}$.

2 Transformation Approach

Friedman's detour can be generalized by considering a transformation $T: \mathcal{R} \rightarrow \mathcal{R}$ on the random variable X so that $Y = T(X)$. Then if X has distribution function $F(x)$ and density $f(x)$, let Y have distribution function $G(y)$ and density $g(y)$. Given that the null density of $f(x)$ is $\phi(x)$ we denote the null density of $g(y)$ to be $\psi(y)$. The transformation, T , can be considered to transform the index, I , into a form suitable for estimation by an alternative orthonormal basis, and to adjust its sensitivities to particular structures.

In somewhat reverse logic, now start with I , defined in its transformed state (with L^2 -distance taken wrt $\psi(y)dy$), and map it back through the inverse transformation:

$$\begin{aligned} I &= \int_{\mathcal{R}} (g(y) - \psi(y))^2 \psi(y) dy \\ &= \int_{\mathcal{R}} \left(\frac{f(x)}{T'(x)} - \frac{\phi(x)}{T'(x)} \right)^2 \phi(x) dx \\ &= \int_{\mathcal{R}} (f(x) - \phi(x))^2 \frac{\phi(x)}{(T'(x))^2} dx \end{aligned}$$

This form clearly shows that the index is a weighted distance between $f(x)$ and a standard normal density, with weighting function $\phi(x)/(T'(x))^2$.

Using this formulation the Legendre index, I^L , (1.1) becomes:

$$I^L = \int_{\mathcal{R}} (f(x) - \phi(x))^2 \frac{1}{2\phi(x)} dx,$$

since $T(X) = 2\Phi(X) - 1 \Rightarrow T'(X) = 2\phi(X)$. Ironically the mapping proposed by Friedman to reduce the influence of tail fluctuations does exactly the opposite. The $1/\phi(x)$ effectively upweights tail observations leaving the Legendre index very sensitive to differences from normality in the tails of $f(x)$.

Through different considerations Hall (1989) observed the same phenomenon. It motivated him to propose an alternative index that measures the L^2 -distance between

$f(x)$ and the standard normal density with respect to Lebesgue measure:

$$I^H = \int_{\mathcal{R}} (f(x) - \phi(x))^2 dx$$

This index, interestingly, can be obtained through a transformation as well. Equating the implicit weight 1 with $\phi(x)/(T'(x))^2$ we find $(T'(x))^2 = \phi(x)$, or $T'(x) = \phi^{1/2}(x)$ and hence $T(X) \propto \Phi_{\sigma=\sqrt{2}}(X)$. Such a transformation seems unnatural, and perhaps a better understanding for what it means is obtained by noticing that I^H gives equal weight to all differences along \mathcal{R} between $f(x)$ and $\phi(x)$. Aside from this, Hall's motivation for the design of the index is from an established approach in density estimation.

We return then to Friedman's original idea of giving more weight to differences in the center, which can be achieved by letting $T(X) = X$, the identity transformation, giving

$$I^N = \int_{\mathcal{R}} (f(x) - \phi(x))^2 \phi(x) dx$$

We call this index the Natural Hermite index, and Hall's index the Hermite index because both use Hermite polynomials in the expansion of $f(x)$, but I^N is "natural" since the distance from the normal density is taken with respect to normal measure.

3 Density Estimation

In each of these indices $f(x)$ (or $g(y)$ in the transformed version) is expanded using an orthonormal functions, as follows, $f(x) = \sum_{i=0}^{\infty} a_i p_i(x)$.

In the Natural Hermite index, $\{p_i(x), i = 0, 1, \dots\}$ is the set of standardized Hermite polynomials orthonormal with respect to (o.n. wrt) $\phi(x)$. (Note that $\phi(x)$ is also called the weight function of the polynomial basis.) More specifically, $p_i(x) = (i!)^{-1/2} H_{e_i}(x)$ where $H_{e_i}(x)$ is a Hermite polynomial defined by the recurrence relation, $H_{e_i}(x) = xH_{e_{i-1}}(x) - (i-1)H_{e_{i-2}}(x)$, $H_{e_0}(x) = 1$, $H_{e_1}(x) = x$. (The subscript "e" is a convention used to distinguish this Hermite polynomial basis from the basis o.n. wrt $\phi^2(x)$.)

In addition, $\phi(x)$ is expanded as $\sum_{i=0}^{\infty} b_i p_i(x)$. Inserting both of these expansions into I^N gives,

$$\begin{aligned} I^N &= \int_{\mathcal{R}} \left(\sum_{i=0}^{\infty} (a_i - b_i) p_i(x) \right)^2 \phi(x) dx \\ &= \sum_{i=0}^{\infty} (a_i - b_i)^2 \text{ since } p_i \text{'s o.n. wrt } \phi(x) \end{aligned}$$

where $a_i = \langle f(x), p_i(x) \rangle_{\phi(x)dx}$,

$b_i = \langle \phi(x), p_i(x) \rangle_{\phi(x)dx}$ ($= a_i$ when $f \equiv \phi$).

Because a_i depends on f , it is unknown and must be estimated in order to estimate I^N . We can rewrite a_i in integral notation as $a_i = \int_{\mathbb{R}} p_i(x)\phi(x)f(x)dx \equiv \int_{\mathbb{R}} p_i(x)\phi(x)dF(x)$, which can be reinterpreted as an expectation, $a_i = E_F\{p_i(X)\phi(x)\}$. In practice, a_i is estimated from the data by a sample average, $\hat{a}_i = \frac{1}{n} \sum_{j=1}^n p_i(x_j)\phi(x_j)$, and then I^N is estimated by using \hat{a}_i and truncating the sum,

$$\hat{I}_M^N = \sum_{i=0}^M (\hat{a}_i - b_i)^2$$

Notably the truncation at M constitutes a smoothing of the true index, I^N .

The approximations in both Legendre and Hall's Hermite indices are similarly constructed. In the Legendre index, the set $\{p_i(x), i = 0, 1, \dots\}$ becomes the set of standardized Legendre polynomials. Hall's Hermite index uses Hermite polynomials o.n. wrt $\phi^2(x)$ defined by the recurrence relation $H_i(x) = 2xH_{i-1}(x) - 2(i-1)H_{i-2}(x)$ with $H_0(x) = 1, H_1(x) = 2x$.

4 Structure Detection

Using the Natural Hermite index as an example, in projection pursuit we are interested in maximizing $\hat{I}_M^N = \sum_{i=0}^M (\hat{a}_i - b_i)^2$ over all possible projections. Light can be shed on this by examining the maximization of the components $(\hat{a}_i - b_i)^2$. Being a square in \hat{a}_i each component is maximized by either minimizing or maximizing \hat{a}_i . When \hat{a}_i is rewritten as $E_F\{p_i(X)\phi(X)\}$ it shows the problem reduces to finding the types of distributions, $F(x)$, which minimize or maximize the expectation. In practice, $F(x)$ is restricted to the set of distribution functions of all 1-dimensional projections of \mathcal{Z} , however to understand the general types of distributions to which a_i responds consider $F(x)$ belonging to an expanded set, $\mathcal{F} = \{F(x) : EX = 0, \text{Var}X = 1\}$.

To consider minimizing or maximizing a_i over \mathcal{F} one also needs to consider the limit points of \mathcal{F} which may not satisfy the variance constraint. Also it can be shown that the distribution which minimizes or maximizes each a_i has at most three point masses, which makes the subsequent analysis simpler.

4.1 Truncation at First Term

Consider the simplest case, truncation of the sum at $M = 0$, $\hat{I}_0^N = (\hat{a}_0 - b_0)^2$. In our implementation of projection pursuit we found that the Natural Hermite index truncated at 0 is particularly valuable for finding large structure, such as clusters, easily. Since $p_0(x) = 1$, $a_0 = \int_{\mathbb{R}} \phi(x)dF(x)$, and $b_0 = 1/(2\sqrt{\pi})$.

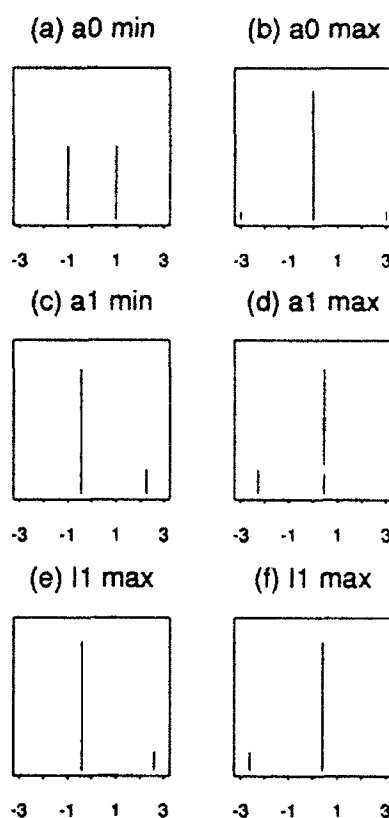


Figure 1: Distributions which maximize Natural Hermite index, (a) minimizes a_0 , (b) (nearly) maximizes a_0 , (c) (nearly) minimizes a_1 , (d) (nearly) maximizes a_1 , (e), (f) (nearly) maximize \hat{I}_1^N .

Proposition:

(i) a_0 is minimized at $1/\sqrt{2\pi e}$, by a distribution with equal mass of size 0.5 at ± 1 . (Call this distribution type I, or a "central hole").

(ii) a_0 is maximized at $1/\sqrt{2\pi}$, by a degenerate distribution with all mass at 0. It also maximizes \hat{I}_0^N at $(\sqrt{2} - 1)^2/4\pi$. (Call this distribution type II, or "heavy central mass").

Proof:

$$\begin{aligned} \text{(i) } a_0 &= \frac{1}{\sqrt{2\pi}} Ee^{-\frac{1}{2}X^2} \\ &\geq \frac{1}{\sqrt{2\pi}} e^{-\frac{1}{2}EX^2} \text{ by noting that } e^{-a} \text{ is convex,} \\ &\quad \text{and Jensen's inequality} \\ &= \frac{1}{\sqrt{2\pi}} e^{-\frac{1}{2}} \text{ by the mean/variance constraints} \\ \text{(ii) } a_0 &= \frac{1}{\sqrt{2\pi}} Ee^{-\frac{1}{2}X^2} \\ &\leq \frac{1}{\sqrt{2\pi}} \text{ because } e^{-\frac{1}{2}X^2} \text{ is monotonically} \\ &\quad \text{decreasing in } X^2 \end{aligned}$$

Intuitively (i) says that to minimize $E_F \phi(X)$, $F(x)$ should have mass as far out in the tails of R as possible, but the mean and variance constraints restrict this mass to be no further out than ± 1 . (This type I distribution is plotted in Figure 1(a).) Conversely (ii) says that because $\phi(x)$ has one maximum at $x = 0$, to maximize $E_F \phi(X)$, $F(x)$ needs all mass at 0. This does not satisfy the variance constraint but it is easy enough to see that is a limiting point of any sequence of distributions in \mathcal{F} with most mass near 0 and small amounts of mass in the tails. A member of such a sequence is plotted in Figure 1(b).

Practically this behaviour means that I_0^N will respond to projections with either heavy central mass and scattered outliers or central holes. The behaviour of I_0^H is identical but unfortunately the Legendre index doesn't have an equivalent term. In fact $I_0^L = 0$, for all $F(x)$.

4.2 Truncation at Second Term

4.2.1 Second Term alone

The second term is a_1^2 where $a_1 = \int_{\mathcal{R}} x \phi(x) dF(x)$ since $p_1(x) = x$, and $b_1 = 0$. In this case it can be shown that the set of distributions which maximize a_1 have mass at only two points. Given this, fix $x_0 > 0$ to be the position of the largest point mass, then the constraints give the size of the mass at x_0 to be $1/(1+x_0^2)$, and the second point mass will be of size $x_0^2/(1+x_0^2)$ at $-1/x_0$. Then taking first and second derivatives of a_1 with respect to x_0 gives the maximum ($\doteq 0.122$) at $x_0 \doteq 0.439$. This is a right-skewed distribution, and the reflection of this distribution exhibiting left-skew will minimize a_1 at $\doteq -0.122$. (These are plotted in Figure 1(c),(d).) So the second term detects skewed distributions. (Call these distribution type III).

This behaviour is essentially the same for both Hermite and Legendre indices.

4.2.2 Piecing Components Together

Truncating the sum at two terms gives the order 1 index, $\hat{I}_1^N = (a_0 - 1/(2\sqrt{\pi}))^2 + a_1^2$, so the behaviour of \hat{I}_1^N depends on the interactive behaviour of the two terms. The distribution which maximizes \hat{I}_1^N is of type III but it is drawn more towards the type II distribution than the type III distribution which separately maximizes the second term, because of the contribution from the first term. This is pictured in Figure 1(e),(f).

Practically, then, the order 1 index responds to skewness in the data when it is present, otherwise its behaviour will be the same as the order 0 index. The Hermite index will behave much the same, but the Legendre index will respond only to skewness because it lacks a contribution from the first term.

4.3 Higher order indices

There is a general trend for odd order indices to be maximized by skewness, type III distributions, and even order indices to be maximized by heavy central mass, type II distributions, but as the order increases the type II distributions tend to dominate both. However this global behaviour is not as interesting as it was for the low order indices because their real power in detecting finely structured projections stems from their increased multimodality.

There is a big difference between the three indices in the speed at which they can detect fine structure. The Legendre index needs considerably fewer terms to collect the same intricacy as do the Hermite indices. A rationale for this is given by asymptotic results in Hall (1989).

4.4 Two-dimensional Indices

The behaviour exhibited by the 1-dimensional indices extends fairly directly to 2-dimensional indices. The low order indices have the "myopic" quality of simply and easily finding large structure from a distance whilst the higher order indices are fine tuners to intricate structure. The order 0 Hermite indices are attracted to central holes because they are maximized by a distribution with all mass placed symmetrically on a unit circle. The order 1 indices respond to bivariate skewness.

5 Illustrations of Projection Pursuit

To illustrate the 1-dimensional index behaviour we use data from Lubischew (1962). There are 6 physical measurements on 3 species of flea beetles. To construct 2-dimensional data we take the 4th and 5th variables, and in this view the clusters are separated but closely knit and the points lie on parallel lines because the 5th variable contains discrete measurements. On this data we conducted 1-dimensional projection pursuit with indices of order 0, 1 and 25. For the analysis a set of projection vectors has been generated by taking $\alpha_\theta = (\cos \theta, \sin \theta)$, $\theta = 0^\circ, 1^\circ, \dots, 179^\circ$. The 2-dimensional data is projected into each α_θ and the index value is calculated. Referring to Figure 2, the data is plotted in the center of the figure and the index value is plotted in relative distances from the center, in both positive and negative directions along the relevant projection vector, α_θ . The maximum value is denoted by "x" and the dotted circle is a guidance line plotted at the median index value. In plot (a) \hat{I}_0^N is calculated for each 1° incremental projection, and (b) contains a histogram of the projected data corresponding to the maximum index value. It is interesting to see that this has a dip in the center and relatively equal mass on either side near ± 1 , approximating a type I distribution. Plot (c) shows \hat{I}_1^N ,

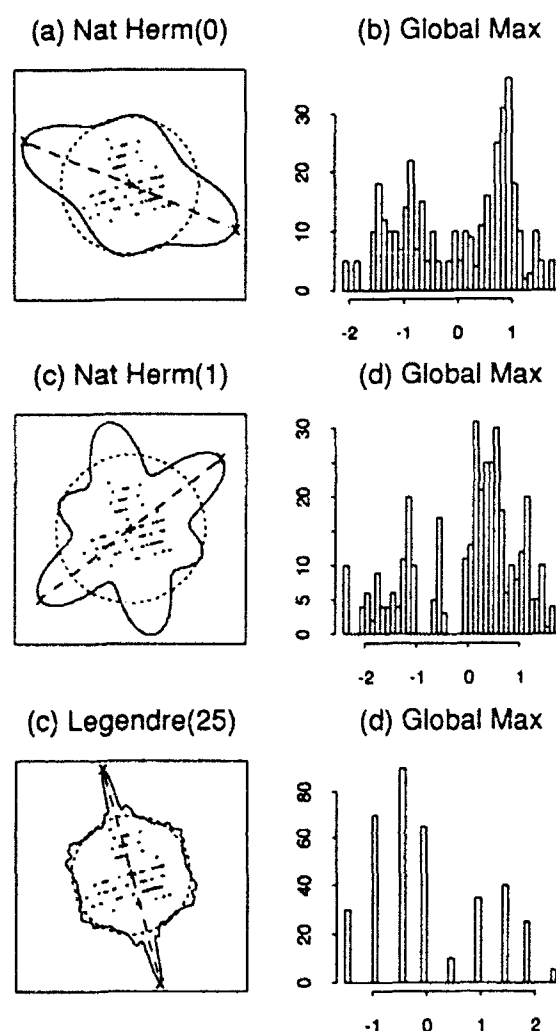


Figure 2: Flea beetles (a) I_0^N , (b) Projection which maximizes I_0^N , (c) I_1^N , (d) Projection which maximizes I_1^N , (e) I_{25}^L , (f) Projection which maximizes I_{25}^L .

order 1 index, calculated for each 1° incremental projection. Contrary to \hat{I}_0^N this is maximized by a projection that separates off one cluster, that is, has large mass to one side and smaller mass on the other side, approximating a type III distribution, as seen in the histogram in plot (d). The two other local maxima each separate off one of the other clusters, but not as clearly as the global maximum. This example illustrates the "myopic" power of the order 0 and 1 indices for separating. An order 0 index will capture projections with two relatively equal clusters whilst an order 1 index will capture projections with one large cluster and a smaller cluster (perhaps obtained by projection of 2 clusters on top of each other and a single cluster, as in this example). To illustrate the sensitivity of higher order indices to fine structure

we switch to the Legendre index. In contrast to the order 0 and 1 indices the global maximum of the order 25 Legendre index, \hat{I}_{25}^L , corresponds to the discrete measurements. (Note the increased modality means a severe tradeoff in smoothness, though.)

6 Discussion

A few related matters are worth mention. One is that the terms in the expansions are themselves measures of non-normality and rate as reasonable indices for specific structure in their own right. Being specific in their responsiveness makes them interesting in practice: one knows what one is looking for. Another matter concerns the introduction of wavelets as a tool for density estimation, and these like the polynomials are fast to compute, and have more tailoring capacity.

7 References

- [1] Cook, D., Buja, A., Cabrera, J. (1991) Direction and Motion Control in the Grand Tour. *Proc. of the 23rd Symp. on the Interface between Comput. Sci. and Statist.*, Springer-Verlag, New York.
- [2] Friedman, J. H. and Tukey, J. W. (1974) A Projection Pursuit Algorithm for Exploratory Data Analysis. *IEEE Trans. Comput. C* 23 881-889.
- [3] Friedman, J. H. (1987) Exploratory Projection Pursuit. *J. Amer. Statist. Assoc.* 82 249-266.
- [4] Hall, P. (1989) Polynomial Projection Pursuit. *Ann. Statist.* 17 589-605.
- [5] Huber, P. J. (1985) Projection Pursuit (with discussion). *Ann. Statist.* 13 435-525.
- [6] Huber, P. J. (1990) Data Analysis and Projection Pursuit. *Tech. Report No. PJH-90-1*, M.I.T.
- [7] Jones, M. C. (1983) The Projection Pursuit Algorithm for Exploratory Data Analysis. PhD Thesis, University of Bath.
- [8] Knuth, D. E. (1981) *The Art of Computer Programming. Vol. 2 Seminumerical Algorithms*. Addison-Wesley, Reading, Mass.
- [9] Kruskal, J. B. (1969) Toward a practical method which helps uncover the structure of a set of observations by finding the line transformation which optimizes a new "index of condensation". In *Statistical Computation*, (R. C. Milton and J. A. Nelder, eds), 427-440. New York: Academic Press.
- [10] Lubischew, A. A. (1962) On the Use of Discriminant Functions in Taxonomy. *Biometrics* 18 455-477.
- [11] Swayne, D. F., Cook, D. (1990) XGobi: A Dynamic Graphics Program Implemented in X with a Link to S. *Proc. of the 22nd Symp. on the Interface between Comput. Sci. and Statist.*, Springer-Verlag, New York.

Maximum Entropy Density Estimation using Random Tessellations¹

Leonard B. Hearne and Edward J. Wegman

Center for Computational Statistics, George Mason University, Fairfax, Virginia

Abstract

A tessellation of the support for a probability density estimator is developed as a set of minimum tiles in a d -dimensional real product space. These minimum tiles are defined by $d + 1$ elements in a set of observations. A class of estimators is then presented that maximizes the entropy or information in a density estimate, given a tessellation based on minimum tiles. This class of estimators is shown to be consistent.

1. Introduction

The development of density estimation procedures on observation spaces of dimension greater than one or two has been pursued by many investigators. The technique that is presented here owes much of its intellectual heritage to work in kernel density estimation, binning methods, and data directed random tessellation. The development of kernel density estimation procedures, [Rosenblatt, 1956], [Parzen, 1962], and the extension of these techniques to dimensions greater than one in the 1960 and 1970s by many authors, gave statistics a technique that could be employed for estimating probability densities on finite dimensional spaces. Inherent in these techniques is the assumption of smoothness of the class of estimators, which is controlled by the choice of the kernel weighting function, and the size of the weighting window. More computation efficient techniques for density estimation on multidimensional observation spaces have been proposed by [Scott, 1985], and [Carr et al., 1987]. These techniques use fixed size and location bins, and estimate the density by the proportion of observations in a given bin to the total number of observations. These binning techniques can then be refined by shifting the location of the bins multiple times and averaging the resulting density estimates. This technique, called an Average Shifted Histogram, has been shown by Scott to converge pointwise to the kernel density estimate for a kernel with constant weight, as the number of

shifts increases. The idea of letting the location of the data define the location of the bins, called data directed tessellation, was developed by [Robertson, 1969] and [Wegman, 1975] for one-dimensional data. In particular, Wegman showed that a class of estimators that is strongly consistent could be found.

Maximum entropy density estimation using random tessellations also builds on, and in many cases extends, a number of the founding principals and results from computational geometry [Preparata et al., 1985] [Edelsbrunner, 1987]. The synthesis of these theoretical results has yielded a technique that provides a strongly consistent density estimator that is more tractable in a computational sense than the kernel density technique, and does not make smoothness assumptions about the true density.

The implementation of this technique would have been much more difficult without the availability of object-oriented program design and implementation languages. In particular, this technique has been implemented in the C++ programming language. [Stroustrup, 1986] The capacity of the language to encapsulate complex computations, particularly those operating on data structures, or objects, that are of unknown size a priori, has significantly reduced the time required to implement a reasonably stable computational algorithm.

2. Properties

The technique of maximum entropy density estimation using random tessellations provides a probability density estimate for observations drawn from a finite dimensional observation space. In particular, this technique has application to problems where the observation space can be characterized as a real product space and the support for an estimate can be assumed to be the convex region defined by a set of observations. Many problems in spatial statistics and in other application areas where the basis for the observation space is known to be orthogonal would benefit from

¹This work was supported in part by NSF grant DMS-9002237 and ONR grant N-00014-92-J-1303.

this estimation technique.

The empirical distribution on a finite dimensional space is defined as the amount of probabilistic weight on the subspace of the observation space that is less than some finite dimensional limit vector, divided by the total probabilistic weight of all observations. Between observations, the empirical distribution is constant, where the definition of between will be presented in Section 3. As defined, the empirical distribution does not have a density function, since one can not take a derivative with respect to location. For a distribution function to have a density function the derivative of the distribution with respect to location in the supporting observation space must exist. In that way the integral of the density over a measurable region in the support gives the probability that a random observation will occur in that region.

In the maximum entropy density estimation using random tessellations technique, the density function is assumed constant between observations. The integral of the density is assumed to be a connected manifold in a $(d+1)$ -dimensional space. This assumption gives this technique the general flavor of the empirical distribution. This is discussed in Section 3.

The likelihood function is constructed in Section 6 for the class of estimators that are part of this technique. This function is convex with respect to the assignment of probabilistic weight. The likelihood function is maximized by assigning all probabilistic weight for an observation to the adjacent element of a tessellation that has the smallest integral measure. This may leave elements of the tessellation with no probabilistic weight, and thus they would not be in the support for an estimate. Viewed globally, under maximum likelihood there may be holes in the support for an estimator. If the probabilistic weight for an observation is distributed among all adjacent elements of the tessellation, proportional to the content of adjacent elements, then the difference in the density estimate on adjacent elements of a tessellation will be minimized. Also, all elements of the tessellation of the support will have some probabilistic weight. This assignment of probabilistic weight maximizes the information, or entropy, of an estimate.

The empirical distribution has the property that it is strongly consistent. As the number of observations goes to infinity then the empirical

distribution converges to the true distribution. This is also true for a density that is estimated using the maximum entropy technique. The density estimator converges almost surely to the true density as the number of observations goes to infinity. The proof is sketched in Section 7.

3. Assumptions

There are two fundamental aspects of a probability density function that we will focus on. The first is that there be a relative ordering of observations, such that they may be partitioned into sets based on this ordering. The empirical distribution has this property. The second is that the underlying distribution function have a derivative with respect to location in the supporting observation space. Then the integral of the density over a measurable region of the support will give the probability that a random observation will occur in that region of the support. It is because this property is missing from the empirical distribution that the empirical distribution does not have a density function.

With kernel density techniques both of the above conditions are met. The relative ordering is provided by assuming that the observation space is a real product space. The density has a derivative by virtue of the motion of the kernel weighting function on the supporting observation space. Kernel techniques also implicitly assume additional derivatives from the kernel weighting function. Thus with kernel density estimation techniques certain smoothness assumptions are being made. These are related to the number of derivatives implicit in the density estimator.

Maximum entropy density estimation using random tessellations also assumes that observations come from a real product space in order to give a relative ordering with respect to each element of the basis to a set of observations. Instead of assuming that there is at least one derivative of the density estimator however, it assumes that the density function is constant between observations. This is much the same as the assumption for the empirical distribution function that the distribution is constant between observations.

Some of the geometric concepts and definitions that may be helpful in understanding this technique are the following.

- A d -dimensional observation space can be projected to a flat, or V_d^1 variety, in a $(d+1)$ -dimensional product space.

- A set of observations can be used to define the vertices of a set of polytopes, of positive integral measure, that form a tessellation.
- If a polytope in a d -dimension space has $d + 1$ vertices, then this polytope is called a minimum tile.
- In a d -dimensional space a point x is said to be between a set of $d + 1$ points if x can be represented as a convex combination of the $d + 1$ points in the set.
- If the distribution function, based on a density estimate, is a connected manifold in a $(d + 1)$ -dimension space, and the density estimate is constant between observations, then the support for this manifold must be a tessellation of minimum tiles.

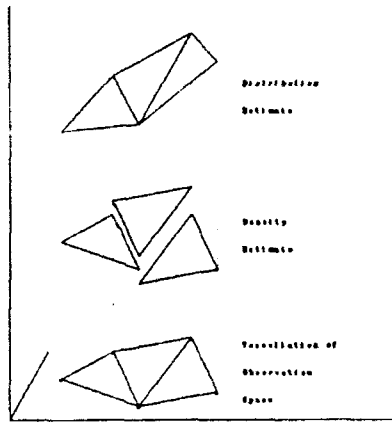


Figure 1

4. Unique Tessellation

To construct a probability density estimate given a set of observations, first construct a tessellation of d -dimensional minimum tiles defined by a set of observations. This is a convex subspace and is the support for an estimate. Each observation is assumed to have some probabilistic mass, and each observation is a vertex for one or more minimum tiles in the tessellation. For each observation assign its probabilistic mass to adjacent minimum tiles in proportion to the integral measure of the adjacent tiles. Then the density estimate on each minimum tile is the probabilistic mass that is assigned to that tile by its $d + 1$ vertices divided by the total probabilistic mass of all observations times the integral measure of the tile.

A tessellation of a d -dimensional observation space, given a set of observations, is not necessarily unique. Consider the observation points in the two dimension space in Figure 2. If points are entered

into the tessellation in vertical rank order and connected to all visible points then the tessellation at the top of the figure results. If instead the points are entered in horizontal rank order then the bottom tessellation results. For an estimate to be unambiguous, given a set of observations, then the tessellation of the support should be unique.

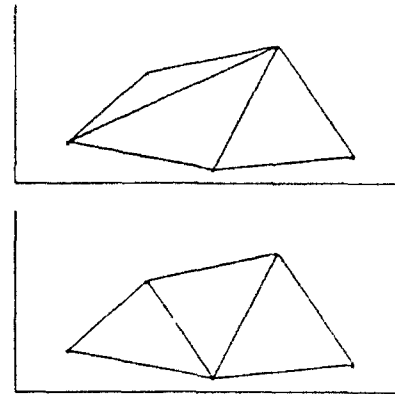


Figure 2

A minimum tile is defined by $d + 1$ points. These same points uniquely define a circumscribed d -dimensional sphere, almost always. If this sphere does not contain any observations in its interior then the minimum tile defined by these $d + 1$ points has the Delaunay property. If all of minimum tiles in a tessellation have the Delaunay property then the tessellation is unique. For n observations in a d -dimensional space, with $d < n$, there exists a unique tessellation that has the Delaunay property. [Preparata et al., 1985]

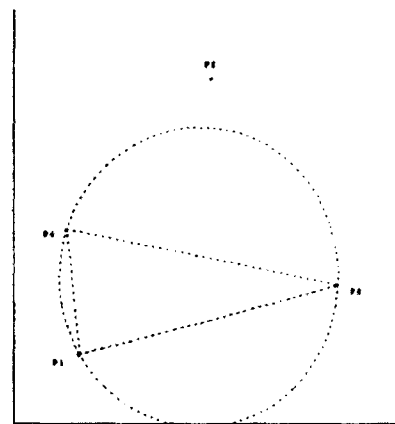


Figure 3

In Figure 3 the spherical cover for the minimum tile formed by the points P_1, P_3, P_4 contains the point P_2 in its interior. So this minimum tile does not have the Delaunay property. The spherical cover of P_1, P_2, P_4 does not contain P_3 , and for the set of four points in two dimensions, this minimum tile has the Delaunay property.

5. A Class of Estimators

Now a class of probability density estimators can now be presented. Let A^n denote the support for an estimator $\hat{f}(x)$, A_i denote an element of the tessellation of the support, $\mu(A_i)$ denote the integral measure of A_i , $W(A_i)$ denote the total probabilistic weight on A_i , and $W(A^n)$ denote the total probabilistic weight of all observations, where n is the number of observations. Then:

$$\hat{f}(x) = \frac{W(A_i)}{W(A^n)\mu(A_i)} I(x)$$

$$I(x) = \begin{cases} 1 & x \in A_i \\ 0 & \text{otherwise,} \end{cases}$$

where m is the number of minimum tiles in the tessellation.

If $W(A_i) = 0$, then $\hat{f}(x) = 0$ and A_i will not be in the support for the estimator A^n . So all A_i must have some probabilistic weight or A^n will not be convex.

6. Likelihood

The likelihood function for this class of estimators can be written in terms of elements of the set of observations:

$$\begin{aligned} L(x) &= \prod_{j=1}^n \hat{f}(x_j) \\ &= \prod_{j=1}^n \frac{W(A_i)}{W(A^n)\mu(A_i)} I(x_j) \end{aligned}$$

Equivalently, the likelihood function can be rewritten in terms of the elements of the tessellating set. Then:

$$\begin{aligned} L(x) &= \prod_{i=1}^m \left(\frac{W(A_i)}{W(A^n)\mu(A_i)} \right) W(A^n) P(x \in A_i) \\ &= \prod_{i=1}^m \left(\frac{W(A_i)}{W(A^n)\mu(A_i)} \right) (W(A_i)/\mu(A_i)). \end{aligned}$$

Since each observation is, by construction, an element in one or more minimum tiles, the amount of probability mass that is assigned to any adjacent tile is variable. If one looks at the second derivative

of the log of the likelihood function with respect to the weight that is assigned to a given tile:

$$\frac{\partial^2}{\partial W(A_i)^2} \ln(L(x)) = \sum_{i=1}^m \mu(A_i)^{-1} \cdot W(A_i)^{-1},$$

it is clear that the likelihood function is convex. To maximize the likelihood, all of the weight of each observation must be assigned to the adjacent tile of minimum integral measure. If one wishes to minimize the variation between adjacent tiles which is the same as maximizing the entropy in the estimate, then the probabilistic weight for each observation should be assigned to adjacent tiles in proportion to the integral measure of the adjacent tiles.

7. Strong Consistency

Let $\hat{f}(x)$ be defined as above. Then $P\left(\lim_{n \rightarrow \infty} \hat{f}(x) = f(x)\right) = 1$.

A sketch of the proof follows.

- Let A be the support for the true density $f(x)$. Given a set of observations $\{Y\}$ drawn from A , then $A^n \subset A$ and $\lim_{n \rightarrow \infty} A^n = A$.

- Let D be the maximum diameter of the spherical covers for all $A_i \in A^n$. Then $\lim_{n \rightarrow \infty} D = 0$.

$$\begin{aligned} \bullet \frac{\text{card}\{Y_i \in \{Y\}: Y_i < x - D\}}{n} &\leq \frac{\int \cdots \int \hat{f}(x)}{A^n < x} \\ &\leq \frac{\text{card}\{Y_i \in \{Y\}: Y_i < x + D\}}{n} \end{aligned}$$

where card denotes cardinality.

Taking the limit as $n \rightarrow \infty$,

$$F_n(x) \leq \lim_{n \rightarrow \infty} \int \cdots \int \hat{f}(x) \leq F_n(x).$$

Under suitable regularity conditions the derivative with respect to location can be taken and

$$f(x) \leq \lim_{n \rightarrow \infty} \hat{f}(x) \leq f(x).$$

Therefore $P\left(\lim_{n \rightarrow \infty} \hat{f}(x) = f(x)\right) = 1$ □

8. Example

Consider the set of four points $\{P_1, P_2, P_3, P_4\}$, in Figure 4, each with unit probabilistic weight. These four points define three minimum tiles $\{A_1, A_2, A_3\} \equiv A^n$, each with the Delaunay property. Let the integral measure of these tiles be: $\mu(A_1) = 1$, $\mu(A_2) = 2$, $\mu(A_3) = 3$.

The attribution of probabilistic weight for each point to individual minimum tiles to give both a

maximum entropy product and a maximum likelihood product is shown in the following table.

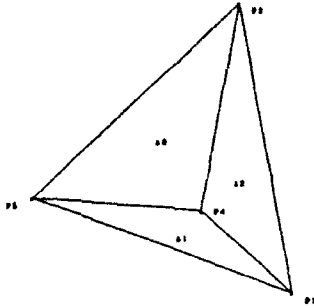


Figure 4

From the table observation P1 assigns 1/3 of its probability mass to tile A1 and 2/3 to A2 under maximum entropy estimation. Under maximum likelihood all of the probabilistic mass is assigned to A1. Repeating this process of all of the observations and then summing the contributions from each observation the total probabilistic weight on each tile can be computed. Applying the results of Section 5 the density estimate for each tile can be computed. Finally, applying the results of Section 6 the likelihood function can be computed under both maximum entropy and maximum likelihood. The likelihood function is larger under maximum likelihood, but at the expense of convexity of the support, since A3 is not an element of the support.

	Maximum Entropy			Maximum Likelihood		
	A1	A2	A3	A1	A2	A3
P1	1/3	2/3		1		
P2		2/5	3/5		1	
P3	1/4		3/4	1		
P4	1/6	2/6	3/6	1		
$W(A_i)$.75	1.4	1.85	3	1	0
$\hat{f}(x)$.188	.175	.154	.75	.125	0
$L(x)$.0266			.0527		

9. Conclusions

Maximum entropy density estimation using random tessellations has been shown to produce a consistent and efficient computational density estimate, that makes no assumptions about the smoothness of the true density. The founding

assumptions for this technique are the density estimate is constant between observations and the distribution is a connected manifold, are much in the flavor of the empirical distribution and lend significant analytic potential to this technique. Some of the areas of direct application of this technique are:

- Density estimation on finite dimensional observation spaces;
- Spatial statistics;
- Analytic treatment of statistical models;
- Estimation of prior and posterior probability densities on finite dimensional spaces.

Bibliography

- Carr, D.B, Littlefield, W.L, Nicholson, W.L, Littlefield, J.S. (1987) "Scatterplot Matrix Techniques for Large N", Journal of the American Statistical Association, V. 82, pp.424-436.
- Edelsbrunner, H. (1987) Algorithms in Combinatorial Geometry, Springer-Verlag, New York: NY.
- Parzen, E. (1962) "On Estimation of a Probability Density Function and Mode", Ann. Math. Stat., V. 33, pp. 1065-1076.
- Preparata, F.P., Shamos, M.I. (1988) Computational Geometry, Springer-Verlag, New York: NY.
- Robertson, T. (1967) "On Estimating a Density Which is Measurable With Respect to a σ -Lattice", Ann. Math. Stat., V. 38, pp.482-493.
- Rosenblatt, M. (1956) "Remarks on some Non-Parametric Estimates of a Density Function", Ann. Math. Stat., V. 27, pp. 832-837.
- Scott, D.W. (1985) "Average Shifted Histograms: Effective Nonparametric Density Estimators in Several Dimensions", Ann. Stat., V. 13, pp.1024-1040.
- Stroustrup, B. (1986) The C++ Programming Language, Addison Wesley, Reading: Ma.
- Wegman, E.J. (1975) "Maximum Likelihood Estimation of a Probability Density Function", Sankhyā: The Indian Journal of Statistics, V. 37, Ser. A, Pt. 2, pp. 211-224.

AN EFFICIENCY STUDY IN NONPARAMETRIC REGRESSION WITH CORRELATED ERRORS

David B. Holiday

UT Health Center at Tyler, P.O. Box 2003, Tyler, TX, 75710

ABSTRACT

Directly optimized weights are used to examine squared error loss for arbitrary linear estimators in a repeated-measures nonparametric regression model which admits correlated-errors. Recent work has shown that uncorrelated-errors lead to symmetric optimality, which matches the form that common kernels induce in equi-spaced designs for kernel estimation. Correlation induces asymmetry in the unrestricted optimal weights and prevents consistency in some models. It is of interest whether restricted optimal estimators, derived under symmetric, normalized, and/or nonnegative constraints, would entail a significant loss of efficiency, and whether asymmetric kernels would be more appropriate for correlated data. The shape of directly estimated effective kernels, not assumed to belong to any particular bandwidth-kernel structure, such as the Gasser-Müller estimator, is explored for range of correlation settings, regression functions, and optimality criteria. Symmetric weights are more efficient than nonnegative/normalized weights. However, a trade-off exists in choosing between them for fixed-designs, depending on whether correlation or regression-function invariance is desired.

1. INTRODUCTION

Hart and Wehrly (1986), Holiday, Wehrly, and Hart (1987), and Holiday (1989, 1991, 1992) have considered the fixed-design nonparametric regression model:

$$Y_s(x_t) = g(x_t) + \epsilon_s(x_t); \quad s = 1, \dots, m; \quad t = 1, \dots, n; \\ \text{cov}\{\epsilon_s(x_t), \epsilon_u(x_v)\} = \delta_{su} \sigma^2 \gamma(x_t - x_v). \quad (1.1)$$

This model is plausible for some growth curves where m independently behaving animals are observed at n times x_1, \dots, x_n , usually equi-spaced, and it is desired to estimate the population average regression g . A parametric or nonparametric positive-definite function $\gamma(\cdot)$ is postulated, which introduces potential correlation among the within-subject errors. The model also generalizes the usual uncorrelated-errors model obtained by setting $m = 1$ and $\gamma(0) = 1$ with $\gamma(u) = 0$ for $u \neq 0$.

Consider a general linear estimator,

$$g_n(x) = \sum_{t=1}^n w_{nt}(x) \bar{y}(x_t) \quad (1.2) \\ \bar{y}(x_t) = \frac{1}{m} \sum_{s=1}^m y_s(x_t), \quad t = 1, \dots, n,$$

which subsumes a large class of kernel, spline, and other estimators. In matrix form, when predicting at points in the design, write

$$g_n = W' \bar{y} \quad (1.3)$$

where W is $n \times n$ weight matrix, and

$$\bar{y}' = [\bar{y}(x_1) \cdots \bar{y}(x_n)], \quad g_n' = [g_n(x_1) \cdots g_n(x_n)].$$

A discrete estimate of the mean integrated squared error, known as the mean averaged squared error (MASE), e.g. Rice (1984), is defined

$$M(W) = E \frac{1}{n} \sum_{t=1}^n \{g_n(x_t) - g(x_t)\}^2, \quad (1.4)$$

and can be expressed as a matrix function of W , the true regression points $g' = [g(x_1) \cdots g(x_n)]$ and the covariance of \bar{y} , which is $\Sigma_m = m^{-1}\Sigma$. Under this model, Holiday (1991) showed

$$M(W) = \frac{1}{n} \text{tr}\{W'(\Sigma_m + gg')W - 2W'gg' + gg'\}. \quad (1.5)$$

with an unrestricted (unr) minimizer of

$$W_{unr} = (\Sigma_m + gg')^{-1} gg'. \quad (1.6)$$

In Holiday (1992), common constraints on the weight matrix in (1.3) were imposed before minimizing, yielding theoretically optimal symmetric (sym), nonnegative (nng), normalized (nrm), and nonnegative-normalized (nnn) weights. Let

$$B \equiv \Sigma_m + gg' = HDH', \quad (1.7)$$

where H is orthogonal and $D = \mathcal{D}[d_1, \dots, d_n]$ contains eigenvalues of B . Then the j^{th} column of the optimal symmetric $W = W'$ is

$$W_{sym} : w_j = 2HD_j H'g, \quad (1.8)$$

where the diagonal matrix D_j is

$$D_j = \left\{ \sum_{s=1}^n \sum_{t=1}^n \frac{g(x_s) h_{jt} h_{st}}{d_t + d_1}, \dots, \sum_{s=1}^n \sum_{t=1}^n \frac{g(x_s) h_{jt} h_{st}}{d_t + d_n} \right\}.$$

For the optimal nonnegative weights, it turns out that each column of the optimal matrix is the solution of a quadratic programming (QP) problem:

$$\min_{w_j} w_j' B w_j + c_j' w_j \quad (1.9)$$

with trivial restrictions,

$$W_{nnj} : \text{ s.t. } w_j \geq 0, \quad (1.10)$$

for $j = 1, \dots, n$; furthermore, B is from (1.7) and

$$c_j = -2 g(x_j) [g(x_1), \dots, g(x_n)]'.$$

The optimal normalized weights use (1.9) with

$$W_{nrm} : \text{ s.t. } (1_n 1_n') \cdot w_j = 1_n, \quad (1.11)$$

where 1_n is an n -vector of 1's. The nonnegative and normalized weights use (1.9) with

$$W_{nnn} : \text{ s.t. } (1_n 1_n') \cdot w_j = 1_n, \quad (1.12)$$

$$w_j \geq 0.$$

These weights obviously are not estimators due to dependence on unknown g and Σ . However, MASE values, evaluated at the minimizers, are approximate efficiency bounds in linear estimation, especially for W_{nnr} . W_{sym} may be a benchmark for equi-spaced designs that use popular kernel estimators with symmetric kernels, such as the Epanechnikov, without bias-reducing boundary modification.

Holiday (1991) demonstrated for the parametric correlation model

$$\gamma(u) = \rho^{|u|}, \quad 0 \leq \rho < 1 \quad (1.13)$$

that $M(W_{nnr}) \neq 0$ for fixed m with $n \rightarrow \infty$, unless $\rho = 0$, precluding the possibility of any mean-square consistent linear estimators when $\rho \neq 0$. Also, the form of (1.6) is generally symmetric only if Σ is a diagonal matrix (uncorrelated errors). The Gasser and Müller (1972) or Priestley and Chao (1972) estimators are traditional kernel smoothers, the latter defining the w_{ij} in (1.3) as

$$W_h : \frac{x_{i+1} - x_i}{h} K\left(\frac{x_i - x_j}{h}\right). \quad (1.14)$$

Both yield symmetric or near-symmetric W for equi-spaced designs and even kernel functions $K(\cdot)$, such as the Epanechnikov. Therefore for uncorrelated-errors models, a symmetric W_h (as a function of the bandwidth h) would seem to naturally do well since it matches the form of the optimal. The main question of this paper is, when symmetry is destroyed by introducing correlation, can one optimize symmetrically and retain efficiency, or if not, would asymmetric kernel functions be suggested?

In addition to the symmetry question, other common constraints, such as nonnegative-normalized, are examined. The estimator of Nadaraya (1964) and Watson (1964) with $K(\cdot) \geq 0$ yields nonnegative columns summing to unity for W_h . This estimator has historically been preferred for the case of random x -points (Chu and Marron, 1991). The behavior and shape of similarly-restricted effective kernels are also of interest for a variety of g and Σ .

2. EFFICIENCY STUDY

For a given m, n , define the efficiency ratios,

$$EFF(sym, unr) = \frac{MASE(W_{sym})}{MASE(W_{nnr})} \times 100\% \quad (2.1)$$

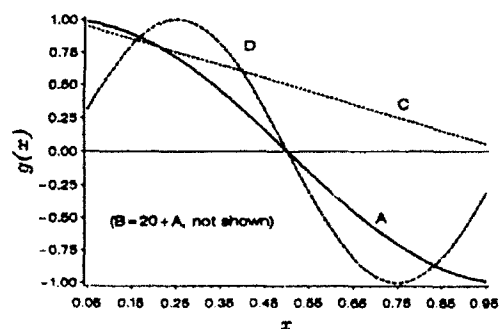
$$EFF(nnn, unr) = \frac{MASE(W_{nnn})}{MASE(W_{nnr})} \times 100\% \quad (2.2)$$

Smaller values imply greater efficiency. Underlying effective kernels for choices of g and a Σ under (1.1) and (1.13) are examined. W_{nrm} , differing little from W_{nnn} , and W_{nnj} , erratic and hard to characterize, were omitted.

For an equi-spaced design, Holiday, Wehrly, and Hart (1987) calculated the minimal $M(W_h)$ with a bandwidth search for W_h induced by the Epanechnikov kernel in the Priestley-Chao, Gasser-Müller, Nadaraya-Watson, and normalized Gasser-Müller estimators, comparing against $M(W_{nnr})$. Severe inefficiencies were noted, although this was prior to knowledge of the restricted minimizers given by (1.7) – (1.12). Some of the same g were chosen here (Figure 1):

$$\begin{aligned} A : & g(x) = \sin(\pi x + (\pi/2)) \\ B : & g(x) = 20 + \sin(\pi x + (\pi/2)) \\ C : & g(x) = 1 - x \\ D : & g(x) = \sin(2\pi x) \end{aligned} \quad (2.3)$$

Figure 1. Regression functions



Other design values were

$$n = 10; \quad x_j = (j - 0.5)/10; \quad j = 1, \dots, 10, \quad (2.4)$$

$$(\sigma^2/m, \rho) \in \{0.1, 1, 10\} \times \{0, 0.2, 0.7\}.$$

An additional sample size ($n = 25$) in Holiday, Wehrly, and Hart (1987) did not yield additional insight, at least for W_{unr} , and was not considered. The ratios (2.1), (2.2) are given in Table 1. Figure 2 displays the effective kernels, i.e., plots of the columns of W , for $\sigma^2/m = 1$, $\rho \in \{0, 0.2\}$. Kernel features were less affected by σ^2/m . Tendencies for increasing ρ are saliently described for only two correlations.

In Table 2, W_{sym} often achieves near 100% efficiency. Interestingly, $M(W_{unr})$ and/or $M(W_{sym})$ lack monotonicity, as a function of ρ (for fixed σ^2/m), for all g except B , which makes general tendencies harder to spot. This apparently does not hold for $M(W_{nnn})$, which steadily increases for all functions. It is easily shown that W_{nnn} is invariant to location shifts in g , as confirmed in Figure 2 for A, B .

The empirical evidence suggests that W_{sym} under this covariance structure is highly invariant to changes in ρ , but not for changes in σ^2/m (data not shown). Since W_{unr} for $\rho = 0$ is inherently symmetric (even for unequally-spaced designs), and thus must equal W_{sym} , there is the implication that W_{sym} can be obtained for any ρ by assuming $\Sigma = \sigma^2 I$. The changing penalty of using W_{sym} when varying $\rho \neq 0$ is not much worse than using the corresponding optimal and asymmetric W_{unr} .

Before W_{sym} is declared the victor, observe W_{sym} for B in Figure 2, which indicates that a near flat constant is the optimum in all cases. Since the curvature in B is the same as A , apart from a constant shift in g , this is unappealing. B , W_{nnn} , and $\rho = 0$ in Figure 2 exhibits the intuitive tendency, with a nice redistribution of weights around the point of estimation as it moves across the interval. In Table 1 for B , W_{nnn} is almost as efficient as W_{sym} in all cases.

Table 1. Efficiencies (2.1), (2.2) for (2.3), (2.4)

$g(x)$	σ^2/m	ρ	$M(W_{unr})$ (1.5, 1.6)	%EFF (2.1)	%EFF (2.2)
A	0.1	0.0	0.0098	100	831
		0.2	0.0172	102	441
		0.7	0.0062	100	1485
	1.0	0.0	0.0833	100	258
		0.2	0.1311	102	543
		0.7	0.0561	100	1639
	10.0	0.0	0.3333	100	400
		0.2	0.3902	100	1588
		0.7	0.2792	100	3187
B	0.1	0.0	0.0110	100	220
		0.2	0.0582	108	130
		0.7	0.0850	105	109
	1.0	0.0	0.1000	100	189
		0.2	0.5781	109	123
		0.7	0.8678	102	106
	10.0	0.0	0.9975	100	134
		0.2	5.7060	108	109
		0.7	8.3074	105	107
C	0.1	0.0	0.0097	100	215
		0.2	0.0313	140	227
		0.7	0.0178	306	513
	1.0	0.0	0.0769	100	189
		0.2	0.1695	115	374
		0.7	0.1202	169	744
	10.0	0.0	0.2495	100	431
		0.2	0.3033	102	1940
		0.7	2.826	109	3074
D	0.1	0.0	0.0098	100	2005
		0.2	0.0094	157	797
		0.7	0.0023	214	4009
	1.0	0.0	0.0833	100	369
		0.2	0.0804	142	863
		0.7	0.0221	201	4136
	10.0	0.0	0.3333	100	400
		0.2	0.3286	111	1868
		0.7	0.1582	148	5601

In Figure 2, the shapes of W_{unr} are similar to g for $\rho = 0$, but become distorted near the boundary as ρ is increased, but apparently less so if g is near 0 at the boundaries, as in D . This is perhaps specific to (1.13). For $\rho = 0$ and linear g , as for C in Figure 2, columns of W_{unr} and of W_{sym} for all ρ (not shown), appear linear as well. Also for C , W_{nnn} kernels appear linear or piecewise linear; for $\rho = 0$, W_{nnn} may vanish at the axis confluence, although this was clearer for another

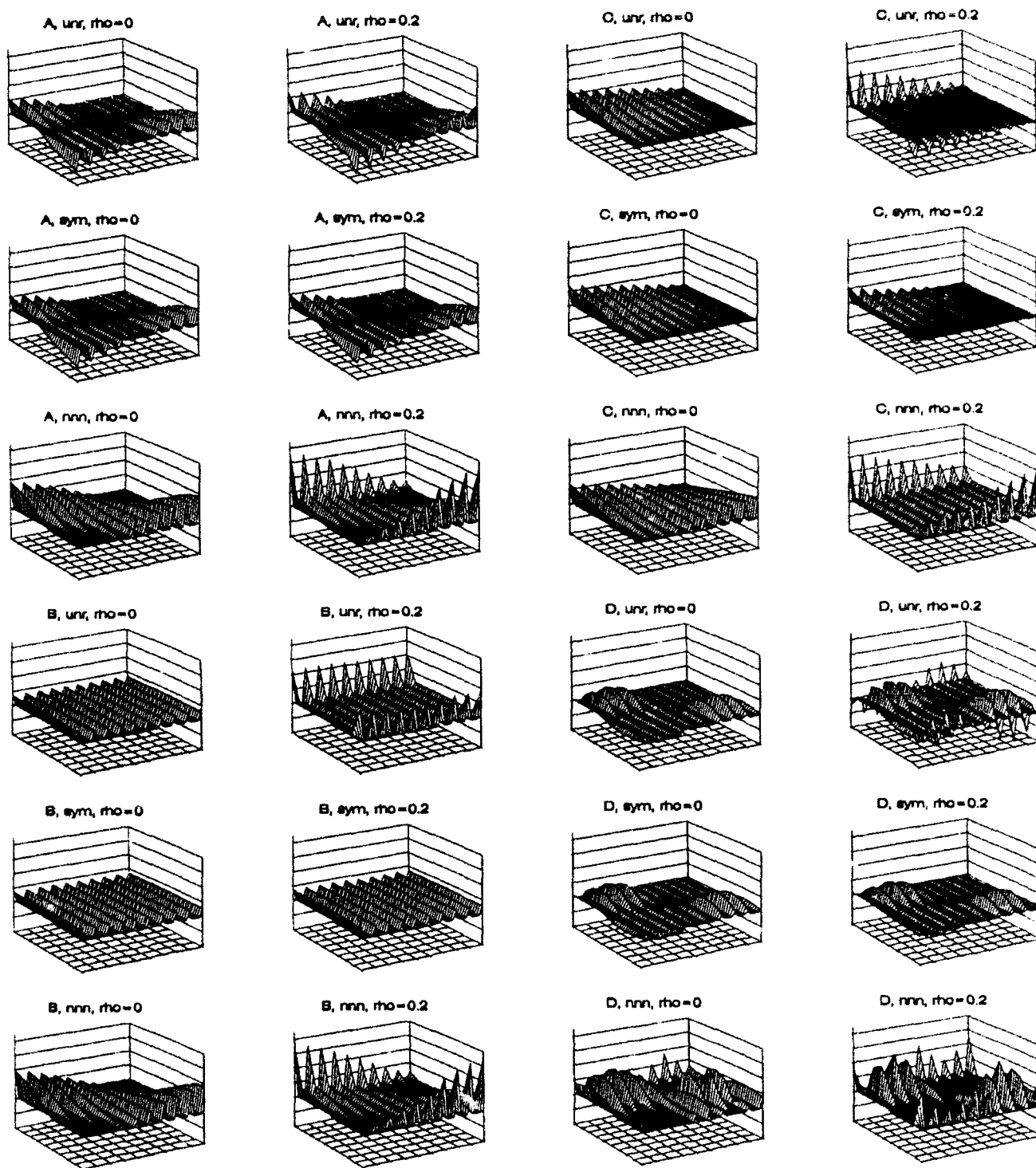


Figure 2. Effective kernels W_{unr} , W_{sym} , and W_{nnn} , given by (1.6), (1.8), and (1.12), respectively. Design is given by (2.3) and abridged (2.4), using only $\sigma^2/m = 1$, $\rho \in \{0, 0.2\}$. Axis description: upright = weights, scaled -0.3 to 0.6 by 0.15 ; lower left = design points, $x_j = (j - 0.5)/10$, $j = 1, \dots, 10$; lower right = columns of W , corresponding to point at which estimation is being made (j).

$g(x) = 2 + 3x$ (not shown); for $\rho \neq 0$, W_{nnn} appears to have three segments, as for C , $\rho = 0.2$ in Figure 2, where the kernels take on a more inverted shape as ρ increases. W_{nnn} kernels are also affected by the size of g at the boundaries, less so for D , as mentioned above.

In Holiday, Wehrly, and Hart (1987), it is argued that the amount of kernel smoothing is governed by the relative sizes of σ^2/m and $\int g^2$. Therefore, when σ^2/m is relatively small or W_{nnn} is used, the percentage of the optimal MASE that is due to variance (remainder due to bias) will be near 100%, indicating that most of the dominating bias has been eliminated. Although W_{nnr} changes a lot for some functions and not others as ρ is varied, this is not terribly important since the more stable W_{sym} achieves almost the same efficacy.

3. DISCUSSION

The use of W_{nnn} in this particular model, although more inefficient, tempers the trade-off for either correlation or regression invariance. Furthermore, W_{sym} would seem to make less sense for unequally-spaced designs. For W_{nnn} and large ρ , it is suggested that a family of asymmetric and potentially inverted kernels, such as that afforded by the generalized beta density, could be useful for interpolating between design points. Modelling the columns of W_{sym} would benefit by using kernels that allow negativity.

The QP problems are easily programmed using commercially available software, and are relatively fast, but require high precision computing; problems tend to occur with hypothetical simulation parameters as often as with real data. These weights have been used to suggest direct estimation schemes, such as a plug-in approach, and is discussed in Holiday (1991,1992). Boundary bias adjustment is automatic.

In summary, the results have yielded insight into the behavior of optimal smoothing under some common restrictions. The issues involved in choosing a traditional kernel regression estimator are admirably covered in Chu and Marron (1991). However, their treatment assumes an uncorrelated errors model, either random or fixed design points, and two historical kernel structures. The theme of the present approach is the form of the various optima should suggest a kernel structure, which is contrary to the usual method of imposing a bandwidth structure and searching for optimal kernels, as in Gasser, Müller, and Mammitzsch (1985). The latter approach has worked well for uncorrelated-errors models, but may be insufficient when correlation is

present. Effects of partially negative correlation models are presently unknown.

ACKNOWLEDGEMENTS

The author would like to thank Jonathan Raz for a critical reading. The technical assistance of Trey Spencer is also greatly appreciated.

REFERENCES

- [1] Chu, C.-K., Marron, J.S. (1991). Choosing a kernel regression estimator. *Statist. Sci.* 6, 404-436.
- [2] Gasser, T., Müller, H. (1979). Kernel estimation of regression functions. In *Smoothing Techniques for Curve Estimation*, Eds. T. Gasser and M. Rosenblatt, pp. 23-68. LNM 757.
- [3] Gasser, T., Müller, H., Mammitzsch, V. (1985). Kernels for nonparametric curve estimation. *J. R. Statist. Soc., B*, 47, 238-252.
- [4] Hart, J.D., Wehrly, T.E. (1986). Kernel regression estimation using repeated measurements data. *J. Amer. Statist. Assoc.*, 81, 1080-1088.
- [5] Holiday, D.B. (1989). The estimation of derivatives of a nonparametric regression function when the data are correlated. *Commun. Statist.-Theory Meth.*, 18, 2107-2124.
- [6] Holiday, D.B. (1991). Considerations for optimal nonparametric regression under a generalized error structure. *Commun. Statist.-Theory Meth.*, 20, 2387-2406.
- [7] Holiday, D.B. (1992). Near optimal weights in nonparametric regression under some common restrictions. *Unpublished manuscript*.
- [8] Holiday, D.B., Wehrly, T.E., Hart J.D. (1987). Considerations for the linear estimation of a regression function when the data are correlated. *Tech. Rep. #3*, TAMU Research Foundation, Project #5321, Department of Statistics, Texas A&M University.
- [9] Nadaraya, E. (1964). On estimating regression. *Th. Prob. Appl.*, 9, 141-142.
- [10] Priestley, M.B., Chao, M.T. (1972). Nonparametric function fitting. *J. R. Statist. Soc., B*, 34, 385-392.
- [11] Rice, J. (1984). Bandwidth choice for nonparametric regression. *Ann. Statist.*, 12, 1215-1230.
- [12] Watson, G. (1964). Smooth regression analysis. *Sankhya, A*, 26, 359-372.

A Hierarchical Genetic System for Symbolic Function Identification

Mingda Jiang and Alden H. Wright
Department of Computer Science
University of Montana
Missoula, MT 59812

Abstract

Given data in the form of a collection of (x, y) pairs of real numbers, the symbolic function identification problem is to find a functional model of the form $y = f(x)$ that fits the data. This paper describes a system for solution of symbolic function identification problems that combines a genetic algorithm and the Levenberg-Marquardt nonlinear regression algorithm. The genetic algorithm uses an expression-tree representation rather than the more usual binary-string representation. Experiments were run with data generated using a wide variety of function models. The system was able to find a function model that closely approximated the data with a very high success rate.

1 Introduction

The function identification problem is to find a functional model of an experimental system, in symbolic form, that fits the experimental data points produced by the system. In this problem, fundamental properties of an experimental system are to be determined from observed behavior of that system. The functional model is a mathematical idealization that is used as an approximate model of the system, and is chosen to give good fit to the given experimental data according to a chosen evaluation criterion.

This paper describes a computer program to solve the one-variable symbolic function identification problem. Researchers in artificial intelligence (machine learning) have investigated the mechanisms of machine discovery and designed some machine learning systems to find empirical laws (function models) from the observations [1][2]. Most of these methods vary widely by task domains. The purpose of this research is to investigate the feasibility of designing a general-purpose machine function identification system which can automatically build a function model to fit the given experimental data. The new method to solve the function identification problems is to combine a symbolic computing method (the hierarchical genetic algorithm) and a numeric computing method (the Levenberg-Marquardt nonlinear regression algorithm) to build a general-purpose machine function identification system, which can find a highly fit function model to a given sample data set. The machine function identification system searches the space of

function models, dynamically creates new generation of function models using a hierarchical genetic algorithm, and optimizes the coefficients of the function models using the Levenberg-Marquardt nonlinear regression algorithm, and tries to make the function models "best" fit the given sample data points. A system called HGSFI (Hierarchical Genetic System for Symbolic Function Identification) was implemented using this design method. This system does not need any priori knowledge about the system or function model to find a function, in symbolic form, that fits a given sample data points.

2 Nonlinear Regression

Regression analysis is a statistical technique for investigating and modeling the relationship between variables. Suppose that we have n pairs of observation data points $(x_1, y_1), (x_2, y_2), \dots, (x_N, y_N)$, and the univariate nonlinear regression equations are as following

$$y_k = f(x_k, B) + \epsilon_k \quad (2.1)$$

where $f(x, B)$ is the known response function, x is an independent variable, $B = [b_1, b_2, \dots, b_M]^T$ is an M dimensional vector of parameters to be estimated, the ϵ_k represents a random error with mean zero and unknown variance σ^2 , and the subscript $k = 1, 2, \dots, N$ ranges over the N observations. The sequence of values of the independent variable $\{x_k\}$ is treated as a fixed known sequence of constants, not as a random variable [3].

The error sum of squares for the nonlinear model and the given data is defined as

$$S(B) = \sum_{k=1}^N [y_k - f(x_k, B)]^2 \quad (2.2)$$

There are several algorithms for minimizing the error sum of squares. The Levenberg-Marquardt algorithm works very well in practice and has become the standard of nonlinear least squares routines [4].

3 Genetic Algorithms

Genetic algorithms are randomized search algorithms based on the mechanics of natural selection and natural genetics [5][6]. A population of potential solutions to the

problem is generated randomly. These potential problem solutions are called individuals, and each individual is represented by a "chromosome". Each individual is assigned a fitness that represents how well it solves the problem. In an optimization problem, the fitness is normally a rescaling of the objective function. In HGSFI, each individual consists of a function model together with values of parameters for the function model.

Most genetic algorithms use a generational approach. The process of producing a new generation involves two major phases. In the selection phase, the more highly fit individual are reproduced, and the less fit individuals are deleted from the population. In the genetic operator phase, genetic operators are applied to the population to generate new individuals. The most commonly used genetic operators are crossover where two individuals are combined to produce one or two new individuals, and mutation, where a single individual is modified in a random way to produce a new individual. It is expected that the genetic operators have some chance of producing individuals that are superior to their parents. The genetic algorithm goes through a sequence of generations.

The most common way of representing the chromosome of an individual is as a bit-string. However, many genetic algorithms that have done well in practice have used a chromosome representation that is tailored to the particular problem.

3.1 Knowledge Representation

To represent various complex function models, we chose the hierarchical structure representation to represent chromosomes (or population members), which was developed by John R. Koza [7]. In a hierarchical structure representation, population members for a particular domain of interest are represented by expression trees. For instance, the cubic polynomial model

$$y = c_1 + c_2x + c_3x^2 + c_4x^3$$

can be represented as the expression tree shown in Fig. 1.

Suppose that in a function identification problem the available set of n functions is $F = \{f_1, f_2, \dots, f_n\}$ and the available set of m terminals is $T = \{a_1, a_2, \dots, a_m\}$. The "terminals" may be variable atomic arguments or constants. Each particular function f in F takes a specified number $z(f)$ of arguments $b_1, b_2, \dots, b_{z(f)}$. Depending on the particular problem of interest, the functions may be standard arithmetic operations such as addition, subtraction, multiplication, and division, and/or standard mathematical functions (such as exp, sin, etc.).

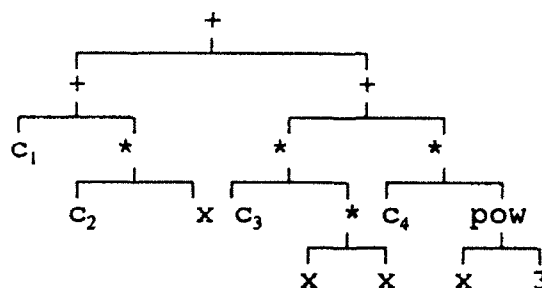


Figure 1: One expression tree for the cubic polynomial model

The solution domain is the set of valid function models that can be recursively created by compositions of the available function operations and the available terminals for the problem. This is represented in the computer as the set of expression trees with a terminal at each leaf and an function operation at each interior node. In our system, the knowledge (i.e. function models) are represented as expression trees.

3.2 Generation of the Initial Population

Genetic algorithms begin with an initial population. Generation of the initial random population in HGSFI begins by selecting one of the functions from the set F at random to be the root of the tree. For an interior node with an k -argument function, k lines are created to radiate out from the node. Then, for each line so created, an element is selected at random from entire combined set $S = F \cup T$ (which is the set of functions and terminals) to be the label for the endpoint of that line. If a terminal is chosen to be any node, the process is then complete for that portion of the tree. If a function is chosen to be the label for any such node, the process continues.

3.3 The Fitness Function

In HGSFI, the Levenberg-Marquardt algorithm is applied to optimize the parameters of each function model before its performance and fitness are evaluated.

In our system, the performance of an individual function model i at generation t is defined as following

$$Perf(i, t) = \frac{1}{1 + \sum_{k=1}^N |y_k - f_i^{(t)}(x_k, B_i^{(t)})|}$$

where $f_i^{(t)}(x_k, B_i^{(t)})$ is the function equation of individual i at generation t . $B_i^{(t)}$ is the estimated parameter vector of

individual i at generation t . The larger the performance, the better an individual in the population. Then, the fitness of individual i at generation t is calculated as

$$f(i, t) = \frac{Perf(i, t)}{\sum_{k=1}^{P_s} Perf(k, t)}$$

where P_s is the population size. The fitness lies between 0 and 1 and is larger for better individuals in the population. The sum of the fitness values in a population is 1. Then the expected number of individuals cloned from an individual in the selection phase is this normalized fitness times the population size.

3.4 Genetic Operators

The genetic operators manipulate individual structures in a population and produce new structures. The two primary operations for modifying the structures undergoing adaptation are crossover and mutation. A third genetic operator, permutation is also used in this paper.

In the crossover operation, individuals in the population are randomly chosen for crossover, employing a user-specified crossover probability. Crossover can be implemented as the following. First, two parents are chosen from the population randomly. Then one tree node in each parent is selected randomly and independently according to a probability distribution. The two offspring are produced by exchanging the subtrees lying below the crossover points. For instance, consider the two parental expression trees in Figure 2.

Assume that the nodes of trees are numbered in a depth-first way starting at the left. Suppose that the seventh node (out of the 17 nodes of the first parent) was selected as the crossover node for the first parent and that the fifth node (out of the 8 nodes of the second parent) was selected as the crossover node of the second parent.

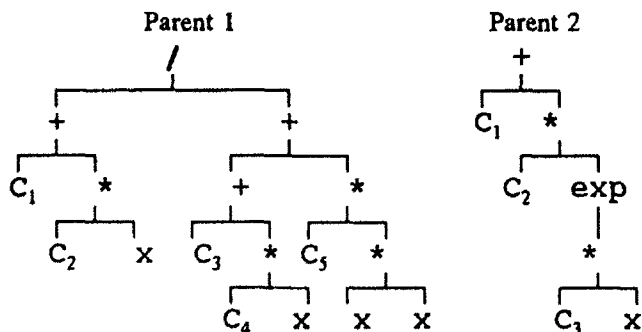


Figure 2: The two parental expression trees

The two offspring resulting from crossover are shown in Figure 3.

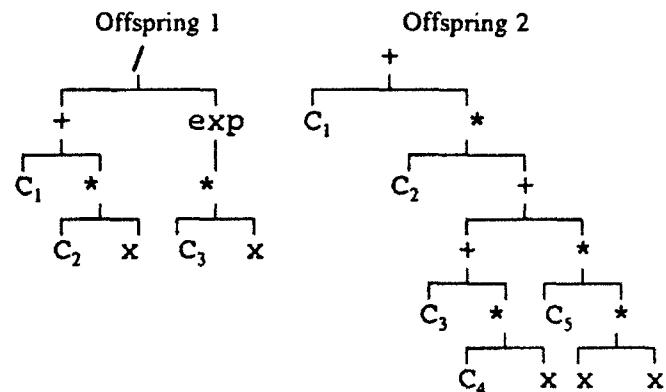


Figure 3: The two offspring resulting from crossover

After crossover, a small number of random nodes of some individuals in the population are selected and changed through mutation. The mutation operation selects an individual from the population and a node of the tree representing the individual at random. Then a member of set S of terminals and functions is chosen to replace the function or terminal at the selected node. If the new function operation has different number of arguments with old function operation at the selected node, the mutation operation needs to add a randomly generated tree branch to the node, or delete the extra branch.

The permutation operation selects an individual from new population, and a function (internal) node of the selected individual randomly. If the function at the selected node is a binary function, then left child and right child are swapped. Note that if the function at the selected node is commutative, there is no immediate effect from the permutation operation on the individual tree.

The kernel of HGSFI is the genetic algorithm. The main loop of HGSFI is an iterative procedure which maintains a constant-size population $P(t)$ of candidate solutions.

4 Experimental Results

We used a wide range of function models from many task domains to test and evaluate the performance of HGSFI machine function identification system. These experiments were run under the UNIX (Ultrix) operating system on a DEC-5500 system.

The test function equations are listed in Table 1. The sample data (no noise) of the test problems were uniformly generated using the test function equations in the given X ranges with the given number of points. For all of the test

problems, we randomly generated initial function models. The only prior knowledge used was the primitive functions, such as +, -, *, /, exp, log, sin, cos, etc., that were used in representing the population of function models. The detail of setting control parameter values to run HGSFI is discussed in [8].

The experimental results for the test problems are listed in Table 2. In the table, the total runs column contains the total number of runs for the given problem. The success rate means the ratio of the number of successful runs to the total number of runs for a given problem. We define a run to be successful for the given sample data set if the convergence test value of some function model in the run

$$C_{test} = 1 - \frac{\sum_{k=1}^N |y_k - f_i^{(t)}(x_k, B)|}{\sum_{k=1}^N |y_k|} \quad (4.1)$$

is greater than the system convergence condition (0.99) within the maximum number of generations (50). The converged generation for a successful run is the generation at which at least one function model satisfies the system convergence condition within the maximum number of generations. The avg. gen. for suc. runs means the average number of the converged generations for successful runs. The CPU time per run is the average CPU time for all runs. The average test values for successful runs and all runs are the averages of the convergence test values (Eq. 4.1) of the best function models for successful runs and all runs, respectively. The average best $R_{squared}$ for successful runs is the average of $R_{squared}$ of the best function models for successful runs.

5 Summary and Conclusions

The HGSFI system combines a hierarchical genetic algorithm and the nonlinear regression algorithm to solve symbolic function identification problems.

On experiments run using data generated using 14 different function models, HGSFI was able to find a function model that closely approximated the data with a high success rate. (Closely approximated meant that the sum of the absolute deviations was less than 1% of the sum of the absolute y values.) Using at least 20 runs per problem, HGSFI succeeded in this sense 100% of the time for 10 of the problems, and 80%, 96%, 97% and 95% of the time for the remaining 4 problems.

HGSFI was less successful at finding the same function

model that generated the data. Especially for complex nonlinear models, HGSFI was likely to find a high-degree polynomial that closely approximated the data [8]. In part, this is because the space of parameterized function models will include many function models that closely approximate a given data set. In addition, a polynomial model that approximates the data reasonably well can be improved by adding on additional high-degree terms with small coefficients.

Limited experiments show that choosing a stricter convergence criterion would improve the ability of HGSFI to find the function model that generated the data. We speculate that a search strategy that forces HGSFI to concentrate on simpler models in early generations before going on to more complex models in later generations would substantially improve the performance of HGSFI in finding the "correct" function model.

References

- [1] Langley, P., Bradshaw, G.L., and Simon, H.A. (1983), *Rediscovering chemistry with the BACON system*, in R.S. Michalski, J.G. Carbonell, & T.M. Mitchell (Eds.), *Machine Learning: An artificial intelligence approach*, San Mateo, CA: Morgan Kaufmann
- [2] Falkenhainer, Brian C. and Michalski, Ryszard S. (1986), Integrating Quantitative and Qualitative Discovery: the ABACUS System, *Machine Learning* 1, 367-401
- [3] Gallant, A. Ronald (1987), *Nonlinear Statistical Models*, New York: John Wiley & Sons
- [4] Press, William H., Flannery, Brian P., Teukolsky, Saul A., and Vetterling, William T. (1988), *Numerical Recipes in C: The Art of Scientific Computing*, New York: Cambridge University Press
- [5] Holland, John H. (1975), *Adaptation in Natural and Artificial Systems*, Ann Arbor, MI: University of Michigan Press
- [6] Goldberg, David E. (1989), *Genetic Algorithms in Search, Optimization, and Machine Learning*, Reading, Massachusetts: Addison-Wesley
- [7] Koza, John R. (1989), *Hierarchical Genetic Algorithms Operating on Populations of Computer Programs*, *Proceedings of the 11th International Joint Conference on Artificial Intelligence (IJCAI)*, San Mateo, CA: Morgan Kaufmann
- [8] Jiang, Mingda (1991), *A Machine Function Identification System Based on Genetic Algorithms*, Master's thesis, Department of Computer Science, University of Montana

Table 1: The set of test problems

Problem	Function Equation	X range	Pts
4.1	$Y = X + X^2 + X^3 + X^4$	[-2.0, 2.0]	100
4.2	$Y = \sin(X) + \cos(X) + X + X^2$	[-5.0, 5.0]	150
4.3	$Y = \cos(X + X)$	[0.0, 6.0]	20
4.4	$Y = X^2 - X^7$	[-5.0, 5.0]	150
4.5	$Y = 3.1416*X + 2.718*X^2$	[-5.0, 5.0]	100
4.6	$Y = 0.808162*\exp(-1.21*X)$	[-2.0, 8.0]	100
4.7	$Y = -2.3 + 3.0*X + 0.45*X^2 - 1.23*X^3$	[-5.0, 5.0]	100
4.8	$Y = 21.10 - 19.81*\exp(-0.00177*X^{3.180})$	[0.0, 14.0]	140
4.9	$Y = 3.2*\sin(X) + 2.5*\cos(X)$	[-2.0, 18.0]	200
4.10	$Y = 0.45 + 6.032*\exp(-X)$	[0.0, 20.0]	100
4.11	$Y = 0.1 - 2.0*X + 0.5*\log(X)$	(0.0, 20.0]	100
4.12	$Y = X / (0.3 + 0.06*X)$	[0.0, 10.0]	100
4.13	$Y = 10.0 / (1 + 25.0*\exp(-0.8*X))$	[0.0, 10.0]	100
4.14	$Y = 0.6 * (e^{-0.3X} - e^{-0.6X}) / (0.6 - 0.3)$	[0.0, 20.0]	100

Table 2: Experimental results of the test problems with LM optimization

Prob-lem	Tot-al run s	Suc-cess rate (%)	Avg. gen. for suc. runs	CPU time per run (min)	Average test value for suc. runs	Average test value for all runs	Average best $R_{squared}$ for suc. runs
4.1	26	100	10.5	37.58	.998411	.998411	.999994
4.2	20	100	10.65	79.33	.998450	.998450	.999983
4.3	30	100	1.066	0.425	.999952	.999952	1.00000
4.4	30	80.0	31.13	84.44	.997754	.923638	.999957
4.5	20	100	3.65	6.94	.999929	.999929	1.00000
4.6	20	100	2.05	3.92	.999880	.999880	.999994
4.7	50	100	10.44	30.10	.998385	.998385	.999992
4.8	49	100	14.31	107.4	.993469	.993469	.999791
4.9	20	100	4.8	16.62	.999776	.999776	1.00000
4.10	23	95.7	10.36	95.75	.999231	.991547	.999993
4.11	25	100	1.0	2.04	.996607	.996607	.999830
4.12	20	100	8.4	99.84	.999674	.999674	.999996
4.13	30	96.7	18.55	350.3	.999344	.998610	.999998
4.14	20	95.0	11.42	152.5	.995469	.994697	.999951

A Study of Neural Network Design and Its Application to Statistical Forecasting

Young R. Park and Chung Chen

School of Management

Syracuse University

and

Thomas J. Murray

School of Industrial Management

New Jersey Institute of Technology

Abstract - A critical issue in neural network design is the determination of an optimal structure of the network for a given task. It is known that the number of hidden units must be sufficient to discriminate each observation correctly. However, a large number of hidden units requires long computational time in training and prediction results may not be reliable.

This study attempts to develop a formal procedure to determine the structure of a multi-layered neural network. The paper applies a statistical model building procedure to neural network design. Principal component analysis (PCA) is proposed to determine the optimal number of hidden units for a multi-layered feedforward network.

I. Introduction

Interest in neural networks has expanded rapidly in recent years. The Parallel Distributed Processing (PDP) research group has greatly contributed to the resurgence of neural network models [13]. Much of the success of neural networks is due to their characteristics of parallel processing, non-linear processing, non-parametric and distributed representation, etc. [9].

While there have been considerable efforts to develop various neural network models, such as feedforward multi-layered networks, recurrent networks, self-organizing maps, etc., and learning algorithms such as the Perceptron and the generalized delta rule, the design for the optimal structure of a network for a given task has still remained a problem. Designing an optimal structure involves finding a structure with the smallest size network which produces the minimal errors for trained cases as well as for untrained cases. The size of the network is defined as the number of parameters to be estimated.

The difficulty in finding the optimal network structure for a task is due to the complex nature of neural network models. Because neural networks are usually non-

linear models and have recursive forms, it is difficult to derive the exact characteristics of the networks.

One common method for structure determination is to increase the number of nodes or layers so that the model can reduce the training errors. Adding nodes or layers may result in fitting the training data closely. However, it introduces a danger that the model will be overfitted. In other words, constructing the network with an excessive number of nodes or layers not only increases processing time but also sometimes loses a generalization capability [17]. Generalization refers to the characteristic that a model can accurately predict untrained data as well as trained data. Therefore, it is crucial to have an objective way to determine the network structure which produces the minimal errors with the smallest size.

Another point to be made here is that most previous studies on selecting the optimal structure for a network have been done in the context of discrete classification problems [4], [15]. However, this paper concentrates on the issue in the context of continuous cases, especially time series forecasting cases.

II. Review of Previous Studies

Previous studies on selecting a network structure can be categorized into four approaches: the ad-hoc approach, the dynamic approach, the distribution approach, and others. The ad-hoc approach means that an experimenter usually has a clear idea of the problem and he/she decides the structure of the network arbitrarily based on his/her experience. The justification for this approach is the robustness of the neural network, that is, the network structure may not affect the performance of the network significantly [16]. Although neural networks are quite robust with a given structure, the structure still makes an impact on the performance. To illustrate this argument, Table 1 shows an experiment done with Sunspot data from 1770 to 1869. As the network model changes the internal structure of the network, the mean square error (MSE) also changes. It

clearly shows that the internal structure of the network affects the performance of the network.

Table 1. Sunspot Data 1770 - 1869
MSE from Each Network Structure

Network Structure	MSE
2 x 5 x 3 x 1	204.27575
2 x 2 x 2 x 1	207.46329
2 x 11 x 1	175.74534
2 x 3 x 1	169.97752
2 x 1 x 1	208.78760
2 x 0 x 1	245.73909

* represents a network without hidden units.

Note 1: All networks are trained with the learning rate 0.3 and the momentum factor 0.7.

Unlike selecting the structure on an ad-hoc basis, the dynamic approach automates the structure selection procedure. The idea is to increase or decrease the number of hidden nodes dynamically during the training process based on system errors. The procedure is repeated until there is no change in the system errors. Typical examples are the method proposed by Hirose *et al.* [6], the cascade-correlation method proposed by Fahlman and Lebiere [3], and the weight-elimination method by Weigend *et al.* [17].

The distribution approach is an attempt to either find or assume the distribution function of estimates; it tries to derive a statistic to measure the performance of the network. For example, White [18] proposes a technique to test whether a current network ignores any non-linear components, i.e., hidden units. Fogel's [4] method is another example for the distribution approach. Under the assumption that the input to the final node has a normal distribution he proposes a statistic called the Final Information Statistic (FIS), which is a derivative of Akaike's information criterion (AIC) [1]. To select the optimal number of hidden nodes, the method requires a calculation of the FIS for each proposed model and chooses the model which has the smallest FIS value. Another technique which belongs to the distribution approach is Sietsma and Dow's pruning approach [14, 15]. Pruning refers to a process that determines unnecessary units in a solution and removes them from a network [14]. Sietsma and Dow divide the pruning process into two stages to eliminate unnecessary units.

Along with the approaches mentioned above, there are some other techniques for this problem. Kung and

Hwang's algebraic projection (AP) method [10] checks the regularity of the training data set and determines the number of hidden nodes based on the regularity. Frean [5] suggests adding a hidden unit whenever a classification error occurs so that the hidden unit can eliminate the error. When there is another kind of error, it adds another hidden unit to control the error.

In summary, although many researchers have made efforts to find the optimal structure for a network, success has been very limited and the results are sometimes contradictory. Still most proposed approaches are on a trial and error basis rather than mathematically proven. In the next section, the paper proposes a method to find the optimal structure for a network.

III. A Method to Determine the Structure of a Network

Structure determination in this paper refers to the questions: (1) how many layers are required and (2) how many nodes are required in each layer. Structure determination does not include the selection procedure of input nodes and output nodes.

As Lippman [11] points out, the two-layered network - a network with only one hidden layer - can describe a fairly complex non-linear relationship as well as a linear one between the input variable and the output variable. Although the three-layered network is more flexible in describing a complicated relationship, it has a drawback, i.e., an increase in processing time. Hornik *et al.* [7] and Hornik [8] prove that the two-layered network with a sufficiently large number of hidden nodes can represent any functional relationship between the input variable and the output variable. Therefore, this study concentrates on the second issue: how many nodes are required in the hidden-layer in the two-layered network.

As mentioned earlier, when a network has a large number of hidden units, it may not produce reliable estimates from the training data and the computation is costly. However, if the network is not large enough, it will fail to describe the relationship between the input variable and the output variable. Therefore, the optimal number of hidden units is important for network design.

III.1. Proposed Technique

The idea of the proposed method is to check whether there is any redundant information on the outputs of the hidden nodes and, if there exists any, then the method eliminates the redundancy by using principal component

analysis (PCA) [12] so that a fewer number of nodes can describe the relationship without losing information. The steps of the proposed method are as follows:

1. Initially train a network with an arbitrarily large number of hidden nodes.
2. Obtain the covariance matrix ($p \times p$) of outputs of the hidden nodes. P is the number of the hidden nodes in the initial network.
3. Apply the principal component analysis technique to the covariance matrix.
4. Obtain the eigenvalues of the matrix.
5. Count the number (p^*) of eigenvalues whose value is greater than 1.
- 6.a. If p^* is less than p , pick p^* nodes out of p by examining the correlation between the hidden nodes and the selected principal components.
- 6.b. Otherwise, it confirms that there is no redundant information. However, it is not guaranteed that the current network has the optimal number of hidden nodes. The system may need more hidden nodes to improve its performance. In this case, the method proposed by White [18] can be used to test whether the system needs additional hidden nodes. However, as long as the initial network has a sufficiently large number of hidden nodes, this seldom happens.
7. After determining the number of hidden nodes, there are two ways to retrain the network with a new structure: one is to randomly generate initial weights again and to retrain the network from the start. The other approach is to use the estimated weights, especially the ones between the selected hidden nodes from step 6.a and the output node as starting values. The second approach is expected to reduce the number of iterations required for training.

In summary, the method examines the covariance matrix of outputs of the hidden nodes in order to test the optimal structure of the network. Each node in the hidden layer has as input the linear sum of the input variables and produces as output the sigmoidal transformation of the input. If there is any redundant information, for example, a hidden node can be represented by another hidden node or a set of hidden nodes, the rank of the covariance matrix will be less than the number of hidden nodes. The advantages of this method are:

1. It is simple and easy to implement.
2. It does not require any assumption on the distribution of estimators.
3. The technique, PCA, is already a well-established technique in multivariate statistics. A drawback of the PCA method is that it is hard to get a meaningful interpretation

for the principal components. In this case, however, the principal components can be considered as the equations to define a hyper-region and p^* is the number of equations to be required to define the region.

4. Unlike other methods which require many experiments with different structures, the proposed method requires training of only two network structures. Another advantage is that it may not require new learning when the principal components are highly correlated with the selected hidden nodes. Because the reduced number of nodes have most of the information possessed by the original nodes, it may not require retraining of the network.

IV. Illustrations

This section illustrates how the proposed method, especially PCA, can be used to select the network structure. An experiment is done with yearly Sunspot data from 1770 to 1869. This series also has been analyzed by Box and Jenkins [2] and is identified as an AR(2) model. A $2 \times 11 \times 1$ network which has Y_{t-1} and Y_{t-2} as inputs is initially trained to predict one-step-ahead forecasted value Y_t . After training the initial network, the covariance matrix of the outputs of the hidden units is calculated to determine the optimal structure of the network. The eigenvalues produced by the matrix are 8.3093, 2.5978, 0.0568, 0.0359, 0.0001, 0.0001, 0, 0, 0, 0, and 0. Two principal components can explain up to 99.15% of the total variation. Therefore, two units are selected as an optimal size for the hidden layer and a network with two hidden nodes is retrained. For comparison the same data set is trained by four other different network models as well. Table 2 shows the MSE from each of these models. All networks are trained with a learning rate 0.3 and a momentum factor 0.7. As illustrated in Table 2, the proposed network has the smallest training error. When the same data set is estimated by the AR(2) model, the MSE from the AR(2) model is 222.95538.

Table 2. MSE from Each Network Structure

Network Structure	MSE
2 x 11 x 1	175.74534
2 x 3 x 1	169.97752
2 x 2 x 1*	163.34526
2 x 1 x 1	208.78760
2 x 0** x 1	245.73909
AR(2)	222.95538

* indicates the proposed optimal structure.

** represents a network without hidden units.

The result shows that the proposed network not only produces the smallest training error among the networks but also has a smaller error than the AR(2) model.

So far, the series have been examined only in terms of training error. However, it is also important for a model to have the capability to accurately predict values outside of the training data set. To examine such a capability of the neural network model, the first 88 observations from 100 observations of the Sunspot data are used for the training data set and the other 12 observations are stored as the testing data set. To compare the power of this model with a time-series model, the AR(2) model is also used for estimation and forecasting with the same data sets. MSE from the AR(2) model for the testing data set is 177.2645 and MSE from the selected neural network model is 163.3360. It shows again that the neural network model has smaller forecasting errors than the AR(2) model for the outside training data set as well.

For this Sunspot data series, it is concluded that the neural network model proposed by this paper produces better forecasted values for the training data set as well as for the outside training data set than the AR(2) model.

Another interesting finding is that when a network has more hidden nodes than the proposed size, the MSE difference is not as significant as when a network has fewer number of hidden nodes than the proposed size. This is consistent with the findings of other neural network research on the so called "robustness" of the structure [16].

V. Conclusions

This paper reviewed and proposed a procedure to design the optimal structure of a neural network model for a given task. Especially, it concentrated on a method to determine the optimal number of nodes in a hidden layer. The empirical example supported the proposed method. When the method is implemented into DSSs (Decision Support Systems), it will help users who may or may not be familiar with neural network models to easily construct a network for a given task. The method proposed in this paper is limited only to the feedforward multi-layered model but the authors believe that the method can be extended to other models as well.

References

- [1] H. Akaike, "A New Look at the Statistical Model Identification", *IEEE Transactions on Automatic Control*, Vol. AC-19, No. 6, December 1974, 716-723.
- [2] G. E. Box and G. M. Jenkins, *Time Series Analysis: Forecasting and Control*, San Francisco: Holden-Day, 1970.
- [3] S. E. Fahlman and C. Lebiere, "The Cascade-Correlation Learning Architecture", *Advances in Neural Information Processing Systems*, edited by Touretzky, David S., Vol. 2, San Mateo, CA: Morgan Kaufmann, 1990, 524-532.
- [4] D. B. Fogel, "An Information Criterion for Optimal Neural Network Selection", *IEEE Transactions on Neural Networks*, Vol. 2, No. 5, September 1991, 490-497.
- [5] M. Frean, "The Upstart Algorithm: A Method for Constructing and Training Feedforward Neural Networks", *Neural Computation*, Vol. 2, 1990, 198-209.
- [6] Y. Hirose, K. Yamashita and S. Hijiya, "Back-Propagation Algorithm Which Varies the Number of Hidden Units", *Neural Networks*, Vol. 4, No. 1, 1991, 61-65.
- [7] K. Hornik, M. Stinchcombe and H. White, "Universal Approximation of an Unknown Mapping and Its Derivatives Using Multilayer Feedforward Networks", *Neural Networks*, Vol. 3, No. 5, 1990, 551-560.
- [8] K. Hornik, "Approximation Capabilities of Multilayer Feedforward Networks", *Neural Networks*, Vol. 4, No. 2, 1991, 251-257.
- [9] K. Knight, "A Gentle Introduction to Subsymbolic Computation: Connectionism for the A.I. Researcher", *Carnegie Mellon Univ. Working Paper, CMU-CS-89-150*, May 1989.
- [10] S. Y. Kung and J. N. Hwang, "An Algebraic Projection Analysis for Optimal Hidden Units Size and Learning Rates in Back-Propagation Learning", *Proceedings of the International Joint Conference on Neural Networks*, Vol. 1, San Diego, CA: 1988, 363-370.
- [11] R. P. Lippman, "An Introduction to Computing with Neural Networks", *IEEE ASSP Magazine*, Vol. 4, No. 2, April 1987, 4-22.
- [12] D. F. Morrison, *Multivariate Statistical Methods*, 2nd edition, New York: McGraw-Hill, 1976.
- [13] D. E. Rumelhart, J. L. McClelland and the PDP Research Group, *Parallel Distributed Processing: Explorations in the Microstructure of Cognition*, Vol. 1, 2,

Cambridge, MA: MIT Press, 1986.

[14] J. Sietsma and R. J. Dow, "Neural Net Pruning - Why and How", *Proceedings of International Joint Conference on Neural Networks*, Vol. 1, San Diego, CA, 1988, 325-333.

[15] J. Sietsma and R. J. Dow, "Creating Artificial Neural Networks That Generalize", *Neural Networks*, Vol. 4, No. 1, 1991, 67-79.

[16] Z. Tang and P. A. Fishwick, "A Comparative Study of the Neural Network and Box-Jenkins Methodology in Forecasting Modelling", University of Florida Working Paper, *UF-CIS-TR-90-13*, June 12, 1990.

[17] A. S. Weigend, B. A. Huberman and D. E. Rumelhart, "Predicting the Future: A Connectionist Approach", *International Journal of Neural Systems*, Vol. 1, No. 3, 1990, 193-209.

[18] H. White, "Some Asymptotic Results for Learning in Single Hidden-Layer Feedforward Network Models", *Journal of the American Statistical Association*, Vol. 84, No. 408, December 1989, 1003-1013.

Smoothing Census Adjustment Factors An Application of High Performance Computing¹

William F. Eddy, Michael M. Meyer, Audris Mockus,
Mark J. Schervish, Kok Tan, and Kert Viele

Carnegie Mellon University, Pittsburgh, PA 15213-3890

Abstract

In this paper we describe a calculation, which was done as part of the Census Post Enumeration Survey, and several computing environments which we used for performing the calculation. The calculation involves much linear algebra and a great deal of storage. Such a calculation should parallelize well. The environments consist of combinations of hardware and languages. We used UNIX workstations, a Cray Y-MP and a Connection CM-2. The languages included Fortran, C, and Linda (a coordination language for distributing Fortran and C programs). We discuss the ways in which the basic calculation is adapted to the different environments, and we compare the environments based on the running time for the calculation.

1 Introduction and Summary

The 1990 Post Enumeration Survey (PES) was designed to produce Census tabulations of states and local areas corrected for under- or over-count. The size of the PES is too small to produce direct estimates for many states and most smaller areas. Therefore estimates were formed for poststrata created by grouping areas across states. Using a statistical model these estimates were smoothed. A detailed description of the methodology can be found in Hogan and Isaki (1991).

The original calculations were written in SAS-IML and performed on a VAX 9000 at the Bureau of the Census. Broadly speaking, we were interested in the costs and benefits of doing these calculations in various other computational environments, particularly in some of the very high-performance environments available to us. This investigation looked at implementations of the same calculations in Fortran and C in several different environments: a Cray-YMP, a Connection Machine CM-2 with 32K processors, and Unix workstations, both individual and collectively, using C-Linda.

¹This project was partially supported by the Bureau of the Census under Joint Statistical Agreement 91-25 with the Department of Statistics, Carnegie Mellon University.

2 The Post Enumeration Survey

After the 1990 Census the Bureau of Census conducted a post enumeration survey—intended to assess the accuracy of the census. The survey looked at approximately 5,300 block clusters and 172,000 housing units. Based on the housing units in the post-enumeration survey there were two evaluations. One, called the P-sample, looked at all people in the sampled blocks and tried to match them with official census records. The other evaluation, called the E-sample, found all census records that should have been in the sample and tried to find the corresponding sample records. There were then two types of error:

1. The first (the undercount) was for the people who were found to be in the sample but not in the census.
2. The second type of error was people who were in the census but not in the sample. These were called erroneous enumerations and are often people whose address is incorrectly recorded or "fictitious" people.

In order to adjust individual census blocks it is important to be able to estimate the undercount (and erroneous enumerations) on a block by block basis. This is clearly not possible, but it is possible to estimate the undercount (and erroneous enumerations) for racial, ethnic, gender, tenure, and age groups which are *a priori* known to differ. This is the concept of post-stratification. Each individual in the PES is placed into one of 1392 post-strata, constructed by crossing 6 age categories with sex and a classification with 115 groups formed using the geographical and other variables plus 12 Native American strata.

3 Statistical Calculations

Raw estimates of the under-/over-count for each stratum are given by simply taking the ratio of the sample count to the Census count. These ratios have a high variability across the various strata, much higher than the actual variability that is expected. Consequently,

the Census Bureau staff have devised various smoothing procedures. We used one as a sample calculation for our exploration. It is described in more detail by Hogan and Isaki (1991).

3.1 Variance Components Model

A variance components model was used to fit the adjustment factors by poststratum as follows. Let

$$Y = X\beta + w + \epsilon$$

where Y is the vector of observed adjustment factors, X is the matrix of explanatory variables, β is the vector of regression parameters, w is the model error assumed to be $N(0, \sigma^2 I)$ independent of the sampling error ϵ assumed to be $N(0, V)$ with V known. The X 's are stratification variables (including interactions); among them are variables reflecting the "degree of census difficulty." In Section 3.2, we explain how the estimates $\hat{\sigma}^2$ and $\hat{\beta}$ are obtained. Once those estimates are obtained, one calculates

$$\begin{aligned}\hat{\Sigma} &= \hat{\sigma}^2 I + V \\ \hat{y} &= X\hat{\beta} + \hat{\sigma}^2 I \hat{\Sigma}^{-1} (Y - X\hat{\beta}) \\ \widehat{\text{Var}}(\hat{\sigma}^2) &= \frac{2}{\text{trace}[\hat{\Sigma}^{-1} \hat{\Sigma}^{-1}]} \\ \widehat{\text{Var}}(\hat{y}) &= V - V \hat{\Sigma}^{-1} V \\ &\quad + V \hat{\Sigma}^{-1} X (X^T \hat{\Sigma}^{-1} X)^{-1} X^T \hat{\Sigma}^{-1} V \\ &\quad + V \hat{\Sigma}^{-1} G \hat{\Sigma}^{-1} V \cdot \widehat{\text{Var}}(\hat{\sigma}^2)\end{aligned}$$

where

$$G = \hat{\Sigma}^{-1} (Y - X\hat{\beta}) (Y - X\hat{\beta})^T \hat{\Sigma}^{-1}$$

The smoothed adjustment factors \hat{y} differ from the regression estimates $X\hat{\beta}$ by a term which is like a fraction of the residual. The fraction used is the ratio between the estimated model error and the total of the model and sampling error.

3.2 Estimation

The procedure by which the parameters of the model are estimated was described by Hogan and Isaki (1991). We summarize their description here. If we assume that V and σ^2 are known, then the generalized least squares estimator of β is

$$\hat{\beta} = (X^T \Sigma^{-1} X)^{-1} (X^T \Sigma^{-1} Y), \quad (1)$$

where $\Sigma = \sigma^2 I + V$ as before. Although V is assumed known, σ^2 is estimated by an iterative procedure. After iteration j ($j = 0$ means before the first iteration) we

have an estimate σ_j^2 of σ^2 and we form $\Sigma_j = \sigma_j^2 I + V$. We then plug this in for Σ in Equation (1) and find $\hat{\beta}_j$. Pretending that $\beta = \hat{\beta}_j$, we find the MLE of σ^2 from $Y \sim N(X\beta, \Sigma)$ and call it σ_{j+1}^2 . This procedure continues until the estimates stabilize. The choice of predictor variables to comprise X was made using a best-subsets regression (Furnival and Wilson, 1974).

The above procedure leads to serious computational difficulties. In the model originally conceived, there are 1392 poststrata. So Σ is a 1392×1392 matrix. For computational reasons (the arrays were too large), separate models were fit to each of four regions (called the regional models). There were 25 poststrata groups in the North East, 38 in the South, 25 in the Midwest, 27 in the West, and 1 on Native American reservations. (Each group is 12 poststrata.) The X matrix contained an average of 20 variables for each of the regional models, and the estimates of σ^2 were around 0.00055.

There were many poststrata with highly variable fitted values, so an alternative model was also fit. This second model is also less computationally intensive. It collapsed the 12 age/sex cross-classifications to 6 categories and removed the Native American group to produce $690 = 115 \times 6$ poststrata. This model is called the national model. There were 14 variables in X in the national model. The matrix V was block diagonal with blocks varying in size from 18 to 66. The estimate of σ^2 was 0.000285 in the national model. The homogeneity within the collapsed poststrata leads to a lower estimated model error.

Totals for the various poststrata were ratio-adjusted so that the national totals would agree with those from the raw data separately for minorities and non-minorities. For example, let c_i be the original census count for poststratum i . If poststratum i is a non-minority poststratum, the fitted value for poststratum i , \hat{Y}_i , is replaced by $\hat{Y}_{i,r} = \hat{Y}_i / r_n$, where

$$r_n = \frac{\sum_{\text{non-minority } j} Y_j c_j}{\sum_{\text{non-minority } j} \hat{Y}_j c_j}.$$

Similarly, a minority poststratum fitted value is divided by

$$r_m = \frac{\sum_{\text{minority } j} Y_j c_j}{\sum_{\text{minority } j} \hat{Y}_j c_j}.$$

The sum of $\hat{Y}_{i,r} c_i$ over all minority or non-minority i will equal the raw sums of $Y_i c_i$ even though the individual Y_i may differ markedly from $\hat{Y}_{i,r}$. The ratio-adjustment factors for the national model were $r_m = 1.00692$ and $r_n = 1.00017$. We used the national model in our computational explorations.

4 Computing Environments

4.1 Hardware

The original computations were done on a VAX 9000 under the VMS operating system. The program was written in SAS-IML, an interactive matrix language extension of the basic SAS system. While the programs are easy to write because of their use of matrix notation, they are not particularly efficient, in part because the programs are interpreted.

4.1.1 Cray Y-MP

The Cray YMP is a multiprocessor pipelined vector machine. It has a similar architecture to the Cray X-MP. The CPU cycle time is 6 nsec, and it has 8 processors. The vectors have length 64. We used both C and Fortran on the Cray but did not attempt to use the multitasking (multiple processors) capabilities.

4.1.2 The Connection Machine – CM-2

The CM-2 system consists of a collection of simple processors, each with its own memory all acting under the direction of a Sun SPARCstation called the front-end. The system depends on *virtual processing*. Virtual processing is a method by which each physical processor simulates some number of virtual processors by subdividing its memory so that each element of data is assigned to a unique processor.

A program is executed on the front-end and that portion of the program which is parallel (as determined by the Fortran-90 array operations which it invokes) is executed on the CM-2. The front end performs all serial operations on scalar data and the CM-2 performs all array operations. When the CM-2 receives an instruction each virtual processor executes the instruction on its own data. Since there is only one set of instructions the processors are naturally synchronized.

The CM-2 has three mechanisms for interprocessor communication:

1. Global communication;
2. Router communication; and
3. Nearest-neighbor communication.

Global communication is used for array reductions, spreading arrays along additional dimensions, and other operations where all processors contribute to the result. Router communication is used for transpositions, permutations, and other array transformations where each processor gets a value from an arbitrary processor. Nearest-neighbor communications is used for shifting data along

array dimensions; each processor gets a value from its neighbor on an n -dimensional grid representing the array.

4.1.3 Workstations

The workstations we used consisted of a collection of DEC UNIX workstations of the following types:

- 3100 MIPS R2000/20MHz 12MB
- 5000/120 MIPS R3000/20MHz 16MB
- 5000/200 MIPS R3000/33MHz 24MB
- 5000/240 MIPS R3000/40MHz 40MB

They were configured to be used by only one person at a time, so they have relatively small primary disks. This means that the amount of space for paging and swapping is a critical constraint on their performance. Communications among the workstation are Ethernet-based with a nominal 10Mb/sec bandwidth.

5 Languages

5.1 Fortran

One concern was the amount of storage which would be needed. There were many large arrays in the SAS code which were freed when no longer in use. The option to free memory is not available in standard Fortran. Instead, we chose to equivalence arrays which would not be needed simultaneously. For example there were 10 arrays of size 690×690 . This would take 38 megabytes if all 10 arrays were to be stored at once. However, no more than 4 of them were needed at once, so they were equivalenced in 4 groups with each group containing one of the 4 which would be needed at the same time. Care was needed to be sure that no 2 arrays in the same equivalence group were needed simultaneously. It turned out that using equivalence made certain of the matrix calculations simpler to program. In several places, each of 21 matrices is manipulated and then the results are appended to each other for future use. For example, the vector $\hat{\beta}$ is multiplied by each of the matrices of independent variables in the regression model to find \hat{Y} s. If the individual matrices of independent variables are each equivalenced to part of an overall independent variables matrix, the multiply operations can all be done in one statement. It was unfortunate for the readability of the code that the independent variable arrays had to be equivalenced in a transposed form due to the way that Fortran stores matrices by columns. This assured that

each of the 21 sets of independent variables would be in contiguous memory locations.

With regard to matrix multiplication, since Fortran stores matrices by columns, care was taken to access the elements of large matrices in a natural order. In particular, since the form $B^T A B$ appeared so often in the calculations, a special routine was written to perform this calculation while accessing both matrices one column at a time. A similar routine was written for $B^T A C$ which also occurred several times.

Several other special purpose routines were written to insert submatrices into diagonal principal minors of a larger matrix, accumulate a sum of matrices, calculate the trace of $A^T A$, and other tasks. Linpack routines were used to solve linear systems (except on the CM-2).

5.2 C

The C port was done independently of the Fortran port. Three main problems had to be solved. First, since swap size was a critical factor, memory had to be conserved. This was done by using the dynamic memory allocation routines *malloc*, and *free*. In order to actually store the matrices in memory, a structure was used which consisted of three fields. These were the number of rows, the number of columns, and a one dimensional array used to store the actual entries. A one dimensional array was used instead of the more natural two dimensional array because of speed considerations. Subsequent tests showed the one-dimensional array took roughly half the time required by the two-dimensional version (presumably because of the simplicity of address calculations). Another structure that was used in the program was an array of 21 matrices. This structure was used because the block diagonal structure of Σ has 21 blocks. Many calculations are therefore done in loops involving these 21 blocks.

The matrix routines from SAS/IML then had to be duplicated in C. Some of these routines, specifically *Invert* and *Solve*, were linked from the Linpack subroutine library. The remainder were written in C. The main problem here was that the one dimensional array involved manual calculation to determine which elements should be used. After the matrix routines were written, the actual translation of the code was fairly straightforward. One problem that arose was that many of the commands in IML involved multiple subroutine calls in the same line. Following the same method in C would have led to huge memory losses, because the memory used to store the intermediate results would have been lost. Therefore each subroutine call had to be done singly, with the intermediate results stored in temporary variables and then freed at the end.

After the translation was finished, a long debugging session was completed. This involved tediously checking intermediate results from the original SAS/IML program with intermediate results from the translated C code. After this was completed, some profiling was done for optimization. Profiling can be done via the *-p* option in the UNIX compiler *cc* together with the UNIX command *prof*. Profiling returns a detailed listing of the time each subroutine in the program is being called. This is useful for identifying subroutines which take excessive amounts of time as well as evaluating the efficiency of changes. A final stage of optimization was done which involved copying the three fields of the record to temporary variables within the linear algebra functions. This reduced the number of pointer references and improved the speed of the program by a factor of 2.

5.3 Porting to the Cray YMP/48

The Fortran code ported relatively easy from workstation to Cray. We only had to change the names of the Linpack routines to the single precision names for the Cray. Some loader directives were needed to enable Fortran to read long (more than 512 characters) records on the Cray.

The C code also needed changes related to the different floating point size on the Cray. The heap (dynamically allocated memory) size had to be specified through a loader directive, as the default size was not big enough. With the increased heap size the program could not be run interactively as it exceeded memory limitations for high priority jobs.

5.4 Porting to the Connection Machine

The Fortran 77 programs can run on the front end computer that is the host for the CM-2. However, they will not use the Connection machine and run as slow as the front end workstation runs. To use the power of the CM-2 special software needs to be used and the program has to be modified substantially.

The software on CM-2 is not as extensive as that for the Cray. To be more exact, a very large portion of any program runs on the front end, but only a small fraction uses Connection Machine itself. There are C-Star and CM Fortran compilers on the Connection Machine as well as Paris directives that could be used from both Fortran and C. C-Star is not as simple to use as CM Fortran and Paris directives are very low level, so only the Fortran version was ported to the CM-2. The port consisted of fully rewriting the Fortran version of the program. One of the reasons was that the workstation version of the program was designed to minimize

the amount of storage and used a number of equivalence statements, as well as specific iterative functions that could not be efficiently translated into array operations that are executed on CM-2. Equivalence statements can not be used for the CM arrays that reside on the Connection Machine.

5.5 Linda

Linda is a coordination language built on top of other languages such as Fortran and C for parallel programming. The Department of Statistics at CMU uses Network C-Linda, the C version of Linda that runs on its network of DECstations for distributed computing.

The advantage of using C-Linda is that its codes are very portable over different parallel computing environments. The end-user need not be modified his code between a shared and a distributed memory environment. Only the compilation and execution procedures need to be modified. Moreover, C-Linda is easy to learn and implement because of the simple set of operations involved, as described later. This does not mean that the programs run efficiently in the differing environments.

C-Linda achieves parallelism by implementing the concept of a "tuple space", a shared memory where data are stored. Processors do not communicate with each other directly, rather data are sent to the tuple space by one processor, and fetched by another. (Of course, this is implemented in our environment by communications between the processors over the Ethernet.)

There are four basic function calls in C-Linda for the operations described above. Data are sent to, or fetched from, tuple space in the form of a *tuple*. A tuple is defined to be a logically ordered set of data. There are two kinds of tuples: data tuples and process tuples. The latter "evaluates" to the former.

As an example, suppose y is some typed variable. Then ("a string", $\exp(7)$, y) is a process tuple that evaluates to the data tuple ("a string", 1096.63..., y).

The four basic calls are :

- 1) `out()`. `out(t)` evaluates a new tuple t and adds it to the tuple space. This is how data are sent to tuple space.
- 2) `in()`. `in(s)` withdraws from the tuple space, some arbitrary tuple t that "matches" the argument s . s is called an *anti-tuple* - a sequence of typed fields which may be actual values or formal place holders. If a field in s has an actual value, then a tuple t must have the same value in the corresponding field to result in a match. If the field is

only a formal place holder, then any t whose corresponding field is of the same type as that in s will do. If no matching occurs, the process invoking `in(s)` will suspend until it does.

As an example, suppose that x is a variable of type float. Then

`in ("a string", ?x, y)`

will withdraw the tuple ("a string", $\exp(7.0)$, y)

from the tuple space.

Here the first and the last field match exactly (i.e., both in type and content), while the second field in `in` matches only the type of the second field in `out` (by the use of the operator '?'). In addition, x is now assigned the value of $\exp(7.0)$.

- 3) `rd()`. `rd(s)` is the same as `in(s)` except that the matching tuple remains in tuple space and is accessible by other processes.
- 4) `eval()`. `eval(t)` is similar to `out(t)`. However, each field in the tuple t is evaluated asynchronously from the invoking process and of each other. This implicitly creates a new process to evaluate each field of t . This is how one spawns multiple processes from the master program to achieve parallelism. (In this case the tuple t is usually a function call.)

The sequential nature of this particular program, and the large amount of memory occupied by the data, limited our success. We were only able to parallelize relatively small and adjacent segments of the code. Unfortunately, these efforts are to disappointing results in the many different attempts made. We outline some of our attempts below.

- (a) In computing $X'\Sigma^{-1}X$, $X'\Sigma^{-1}Y$, and Σ^{-1} . Remember that $\Sigma^{-1} = \sigma^2 I + V$ and note that $\hat{\beta} = (X'\Sigma^{-1}X)^{-1}(X'\Sigma^{-1}Y)$.

Because of the block-diagonal structure of Σ^{-1} , the computation of each of the three quantities above can be reduced to 21 smaller sets of calculation involving the submatrices along the diagonal of V . So we fork 21 sub-processes at this stage for this purpose, and accumulate the results in the appropriate locations in the respective matrices when the processes are complete.

- (b) In computing two sets of large matrix multiplications, each involving two matrices of same dimension (690x690). These two sets of calculation alone take up nearly 1/3 of the elapsed time on a DECstation 5000/240. Because each set is independent

of the other, a slave processor is used to calculate one set while the master does the other at about the same time.

- (c) One slave processor performed the calculation of the four component matrices of $\widehat{\text{Var}}(\hat{y})$ while the master processor did other variance-covariance matrix calculations.
- (d) The "other variance-covariance matrix calculations" in (c) were performed on a second slave processor.
- (e) Separately, we investigated whether it pays to parallelize the matrix multiplication routine itself. The matrix product $C = AB$ can be calculated separately as $C_1 = A_1B$ and $C_2 = A_2B$, where

$$A = \begin{pmatrix} A_1 \\ A_2 \end{pmatrix} \quad \text{and} \quad C = \begin{pmatrix} C_1 \\ C_2 \end{pmatrix}.$$

The largest matrix multiplication in the code involves two matrices of dimension 690x14 and 14x690 respectively, and this happens only twice. On a DECstation 5000/240, 10 repetitions of such a matrix multiplication takes 120.6 seconds. A combination of a DECstation 5000/240 (as the master processor) and DECstation 5000/120 takes 195 seconds of elapsed time, with the master operating at 20% of its capacity only. This shows that the dimensions of the matrices involved are not large enough to compensate for the communication overhead.

The C-Linda codes are modified from the existing C codes. Table 1 gives the results. All times are in seconds, rounded to nearest integer. Usertime is in reference to the master processor. All slave processors were DECstation 5000/120's.

It is obvious that our attempts to achieve a shorter running time have failed. While it may be that a complete rewrite of the program will provide a greater improvement (i.e., a better parallelization), at this point we can only attribute the failures to the large overhead associated with the network implementation of the C-Linda computing environment *relative* to the amount of speed-up that an additional processor or two can offer, for this particular problem.

6 Other Results

Here, we summarize the results of performing the statistical calculation on the various machines in the various languages. Table 2 gives the initial running times before

we made any special efforts to speed up the program. Table 3 gives revised times after these changes had been made.

References

- Furnival, G.M. and Wilson, R.W. (1974). Regression by leaps and bounds. *Technometrics*, 16, 499-512.
- Hogan, H. and C.T. Isaki (1991). The effect of alternative groupings on local area estimates of undercount. *Proceedings of Statistics Canada Symposium 91, Spatial Issues in Statistics*.

types	master	slaves	usertime	totaltime
None	5000/240	0	94	98
None	5000/120	0	189	566
(a)	5000/240	2	93	100
(a)	5000/120	2	189	529
(b)	5000/240	1	79	108
(b)	5000/120	1	160	577
(a+b)	5000/240	3	78	119
(a+b)	5000/120	3	157	544
(c)	5000/240	1	85	104
(c)	5000/120	1	172	555
(c+d)	5000/240	2	80	186
(c+d)	5000/120	2	162	664

Table 1: Running Times in Seconds for C-Linda Version

Compiler Switches	Cray Y-MP	5000/240	5000/120	CM-2
cf77	32.7			
cf77 -Zp -Wf'-o inline -A fast"	33.6			
cf77 -Zv -Wf'-e mcx"	32.8			
cc -d "HEAP=25000000"	44.0			
cc -d "HEAP=25000000" -O2	40.9			
cc -d "HEAP=25000000" -O3	DNC			
gcc -O2 float.c		133.0		
cc -O4 -Olimit 600		111.0	224.2	
gcc float.c		278.3		
cc -O0		235.1		
gcc -O2 -DPRECISE		150.0		
cc -O4 -Olimit 600 -DPRECISE		127.4	354.6	
f77		604.1	1215.4	
f77 -O4 -Olimit 2000		184.9		
cmf (8k)				NEM
cmf (16k) busy				43.2
cmf (16k) elapsed				73.8
cmf -O (16k) busy				37.7
cmf -O (16k) elapsed				65
cmf -O (32k) busy				
cmf -O (32k) elapsed				
f77 -O (front end)				468.2

DNC - program did not compile.

NEM - insufficient memory available

Table 2: Initial Running Times in Seconds

Compiler Switches	Cray Y-MP	5000/240	5000/120
cf77	30.9		
cf77 -Zp -Wf"-o inline -A fast"	30.6		
cf77 -Zv -Wf"-e mcx"	30.9		
cc -DCRAY -d "HEAP=10000000"	14.8		
cc -DCRAY -d "HEAP=10000000" -O2	15.9		
cc -O0		128	261.9
cc -O0 -DPRECISE		139	282
cc -O4 -Olimit 600		51	103
cc -O4 -DPRECISE -Olimit 600		67	135
gcc float.c		191	388
gcc -DPRECISE float.c		206	416
gcc -O2 float.c		58	118
gcc -O2 -DPRECISE		77	162
f77		592	1205
f77 -O4 -Olimit 2000		127	266
f77 -O4 -Olimit 2000(modified)		104	223

Table 3: Revised Running Times in Seconds After Optimization

BOOTSTRAP CONFIDENCE INTERVAL ESTIMATION FOR MODEL-BASED SURVEYS

Kim-Anh Do
Statistical Sciences Division
Centre for Mathematics and Its Applications
Australian National University
Canberra, A.C.T. 2601 Australia

Philip Kokic
Australian Bureau of Agriculture
and Resource Economics
GPO Box 1563
Canberra, A.C.T. 2601 Australia

Abstract

Model-based approaches to survey estimation allow auxiliary population information to be directly incorporated into the estimation process. This auxiliary information is sometimes available for each unit in the population and then is referred to as benchmark information. In multipurpose survey situations when the benchmark variables are almost collinear for the particular sample chosen, a solution to this problem is to resort to a ridge-type model-based method. We consider here the problem of estimating confidence intervals for the population totals where the vector of weights are ridge-type sample weights. Attention is focussed on different methods for setting large-sample confidence intervals about the weighted total estimator \hat{T} of the population total T when the assumed model generating this estimate has misspecified variance structure. We proposed bootstrap methods in constructing confidence intervals and compared these bootstrap confidence intervals to the more conventional ones. We conducted a simulation study comparing these approaches applied to a range of variables collected in the Australian Agricultural and Grazing Industries Survey conducted annually by the Australian Bureau of Agriculture and Resource Economics (ABARE).

1. Introduction

In multipurpose survey situations where the selected sample is unbalanced with respect to important benchmark variables, model-based estimation techniques have been proposed in the face of such problems. These techniques allow for auxiliary population information to be directly incorporated into the estimation process; for example, a linear model can be fitted at the estimation stage of the survey which results in the use of ridge-type sample weights that optimise the trade-off between bias and variance for this situation. See Bardsley and Chambers (1984), Dunstan and Chambers (1986). The problem studied here is that of estimating confidence intervals about population means and totals estimated in this way, in particular when the variance structure in the underlying model is misspecified. Model misspecification can cause considerable bias in the estimation of popula-

tion means or totals and adversely affect coverage probabilities of confidence intervals. Royall and Cumberland (1978) obtained estimators of the error variance of the finite population total estimator which are robust under variance misspecification of the underlying superpopulation model. Dunstan and Chambers (1986) generalised this variance estimator to the case where ridge regression techniques are used to construct the sample weights. Misspecification of the mean structure in the superpopulation model is a significant problem and may also cause considerable bias in finite population estimators. We do not investigate this problem here as its treatment has been thoroughly investigated using techniques such as balanced resampling, see Royall and Herson (1973a, 1973b), and robust methods such as those examined in Chambers (1986). Variance misspecification may be more difficult to detect even though its effects on finite population estimators may be significant. In this paper we compare confidence intervals obtained using the robust variance estimator of Dunstan and Chambers (1986) with the corresponding confidence intervals obtained from the bootstrap. We show that the bootstrap-based confidence interval is considerably more accurate when the variance is misspecified.

2. The Working Model

Suppose the total population \mathcal{P} consists of N distinguishable elements from which a sample S of size n was taken. We systematically use upper case letters to refer to variables defined on \mathcal{P} , and the corresponding lower case letters to refer to their restrictions to S . Let $\mathbf{Y} = (Y_1, \dots, Y_N)^T$ denote the N -vector of population values for a given survey variable and \mathbf{y} is the n -vector restricted to S . We assume that there are k benchmark variables and denote the vector of benchmark variables associated with the i th survey variable as $\mathbf{X}_i = (X_{i1}, \dots, X_{ik})^T$, then $\mathbf{X} = [\mathbf{X}_1, \dots, \mathbf{X}_N]^T$ is the $N \times k$ matrix of benchmark variables for the whole population and \mathbf{x} is the $n \times k$ matrix defined by the restriction of \mathbf{X} to S . For each Y_i ($i = 1, \dots, N$) we assume a model of the form

$$Y_i = \mathbf{X}_i^T \boldsymbol{\beta} + \epsilon_i; \quad E(\epsilon_i) = 0; \quad E(\boldsymbol{\epsilon}^T \boldsymbol{\epsilon}) = \sigma^2 \boldsymbol{\Omega}, \quad (1)$$

where β is a $k \times 1$ vector of unknown variables (depending on Y), and Ω is an $N \times N$ diagonal matrix of known constants Ω_i ($i = 1, \dots, N$) independent of Y (ω is the $n \times n$ analogy of Ω restricted to S).

Let $T(Y)$ denote the unknown population total. The model (1) allows one to model the non-sampled units in terms of the sampled units and benchmark information known for every unit in the population. Using this approach Bardsley and Chambers (1984) constructed the following estimator of $T(Y)$. Specifically, they used

$$\hat{T}(Y) = \sum_{i \in S} w_i Y_i = \sum_{i \in S} Y_i + \sum_{i \notin S} X_i^T \hat{\beta}, \quad (2)$$

where

$$w = 1_n + \omega^{-1} x(\lambda UC^{-1} U + x^T \omega^{-1} x)^{-1} \sum_{i \notin S} X_i, \quad (3)$$

and

$$\hat{\beta} = (\lambda UC^{-1} U + x^T \omega^{-1} x)^{-1} x^T \omega^{-1} y, \quad (4)$$

is the ridge estimator of β and $U = \text{diag}[U_1, \dots, U_k]$ with $U_j = \sum_{i=1}^N X_{ij}$ denoting population benchmark totals, C is a diagonal matrix of non-negative cost coefficients measuring the degree of error we are willing to accept in "estimating" the benchmarks X and λ is a real-valued ridge parameter.

3. The Problem

Given an estimator $\hat{T}(Y)$ of $T(Y)$ satisfying (2) where w is the set of some pre-calculated normalised ridge weights satisfying (4) for known Ω , U , C , X and λ , we are interested in calculating confidence intervals for the population total $T(Y)$ estimated in this way. We have

$$\hat{T}(Y) - T(Y) = \sum_{i \in S} (w_i - 1) \epsilon_i - \sum_{i \notin S} \epsilon_i + b_R, \quad (5)$$

where

$$b_R = \left(\sum_{i \in S} w_i X_i^T - \sum_{i=1}^N X_i^T \right) \beta \quad (6)$$

is the bias term introduced by ridge regression and can be estimated by replacing β with $\hat{\beta}$.

3.1 Simplified robust model-based method

This is the conventional method that ABARE uses to obtain an estimate of the variance of the prediction error as described in Chambers (1986), specifically it is

$$v_C = \left[\left(\sum_{i \in S} w_i^2 - 2 \sum_{i \in S} w_i + N \right) \times \left\{ \sum_{i \in S} w_i^2 - (Y_i - \hat{Y})^2 \right\} \right] / \left[\sum_{i \in S} w_i^2 - 2 \sum_{i \in S} w_i^2 / \sum_{i \in S} w_i + \left[\left(\sum_{i \in S} w_i^2 \right)^2 / \left(\sum_{i \in S} w_i \right)^2 \right] \right] \quad (7)$$

where $\hat{Y} = \sum_{i \in S} w_i Y_i / \sum_{i \in S} w_i$. Hence the simple formula that ABARE uses to construct a $(1 - \alpha)$ -level confidence interval for T is

$$(\hat{T}(Y) - \sqrt{v_C} \Phi^{-1}(\alpha/2), \hat{T}(Y) + \sqrt{v_C} \Phi^{-1}(\alpha/2)).$$

3.2 Dunstan-Chambers method

Dunstan and Chambers (1986) proposed a method of constructing a confidence interval which was robust against variance misspecification in (1). They obtained the formula for $(1 - \alpha)$ -level confidence interval as

$$(\hat{T}(Y) - \hat{b}_R - \Phi^{-1}(1 - \alpha/2) \nu_0^{1/2}, \hat{T}(Y) - \hat{b}_R + \Phi^{-1}(1 - \alpha/2) \nu_0^{1/2}), \quad (8)$$

where \hat{b}_R is an estimator of the ridge bias obtained by replacing β in its formulation with $\hat{\beta}_{WLS} = (x^T \omega^{-1} x)^{-1} x^T \omega^{-1} y$, the best linear unbiased estimator of β under (1), and ν_0 is a robust estimator of $\nu_0 = \text{var}[\hat{T}(Y) - T(Y) - \hat{b}_R]$.

3.3 Confidence intervals based on the simple bootstrap

From (6) we can obtain an expression for the prediction error in terms of the residuals as

$$\hat{T}(Y) - T(Y) = \sum_{i \in S} (w_i - 1) \epsilon_i - \sum_{i \notin S} \epsilon_i + \left(\sum_{i \in S} w_i X_i^T - \sum_{i=1}^N X_i^T \right) \beta. \quad (9)$$

where for $i \in S$ the residuals can be estimated by $\hat{\epsilon}_i = Y_i - X_i^T \hat{\beta}$.

The distribution of $\hat{T}(Y) - T(Y)$ and hence confidence intervals for the population total can be simulated by applying the bootstrap technique to the first term on the RHS of (9), approximating the second term, and finally estimating the third term.

Consider first how to estimate the distribution of $\sum_{i \in S} (w_i - 1) \epsilon_i$ by the bootstrap. Suppose we are resampling residuals where an element $(\hat{\epsilon}^*, w^*)$ in a bootstrap resample is selected at random from $\{(\hat{\epsilon}_i, w_i), i \in S\}$ with uniform probability $p_i = \frac{1}{n}$. This resampling is repeated independently B times and a single bootstrap resample is obtained as

$$R^* = \{(\hat{\epsilon}_1^*, w_1^*), \dots, (\hat{\epsilon}_n^*, w_n^*)\}.$$

The statistic-of-interest here is $R = \sum_{i \in S} (w_i - 1) \epsilon_i$ and its bootstrap analogue is $R^* = \sum_{i=1}^n (w_i^* - 1) \hat{\epsilon}_i^*$.

The exact bootstrap distribution of R^* is in general almost impossible to compute. One way around this problem is to independently replicate R^* many times. This is called the straightforward bootstrap and we label the

B replicated bootstrap values R_1^*, \dots, R_B^* . The distribution of $\sum_{i \in S} (w_i - 1) \epsilon_i$ may then be estimated by the empirical distribution of $\{R_1^*, \dots, R_B^*\}$.

To approximate the distribution of the second term on the right hand side of (9), as the number of non-sampled units is usually very large, it is reasonable to make the approximation

$$\sum_{i \in S} \epsilon_i \approx \text{Normal}\left(0, \sigma^2 \sum_{i \in S} \Omega_i\right).$$

Note that

$$\sigma^2 \sum_{i \in S} \Omega_i$$

may be estimated using $\hat{\tau}$, the robust estimator suggested by Dunstan and Chambers (1986) and referred to earlier in this paper. Thus the distribution of $\sum_{i \in S} \epsilon_i$ may be approximated by a normal random variable with mean zero and variance $\hat{\tau}$.

The third and final term on the RHS of (11) may be estimated by replacing β with the ridge estimator $\hat{\beta}$.

In summary, to construct a $(1-\alpha)$ -level bootstrap confidence interval for the finite population total, we compute R^* , N , a random value from a $\text{Normal}(0, \hat{\tau})$ distribution and compute

$$T^* = R^* - N^* + \left(\sum_{i \in S} w_i X_i^T - \sum_{i=1}^N X_i^T \right) \hat{\beta}.$$

Repeat this independently a large number of times, $B = 50,000$, say, to form B bootstrap values $\tau = \{T_1^*, \dots, T_B^*\}$. Select the $\alpha/2$ and $1-\alpha/2$ quantile values from τ , denoted by $T^*(\alpha/2)$ and $T^*(1-\alpha/2)$ respectively. The simple bootstrap confidence interval is then

$$(\hat{T} - T^*(\alpha/2), \hat{T} + T^*(1-\alpha/2)).$$

3.4 Efficient bootstrap confidence intervals

For the b^{th} bootstrap resample S_b , let \hat{L}_b and \hat{M}_b denote the respective linear and remainder terms where

$$S_b = \hat{L}_b + \hat{M}_b$$

denote the $n \times B$ matrix (P_1, \dots, P_B) . We can correct the linear parts by fitting distributions with appropriate cumulants as follows:

1. Let $(0, \tilde{\sigma}/\sqrt{n}, \tilde{\gamma}/\sqrt{n}, \tilde{\delta}/\sqrt{n})$ denote the cumulant of the empirical distribution of $\hat{L}_1, \dots, \hat{L}_B$ which can be calculated from the resamples.
2. Let $(0, \sigma_\alpha/\sqrt{n}, \gamma_\alpha, \delta_\alpha/\sqrt{n})$ denote the theoretical cumulants of \hat{L} . For a precise calculation of the theoretical cumulants we refer the reader to Efron (1990).

3. We replace the sorted values $\hat{L}_1, \dots, \hat{L}_B$ with $\tilde{L}_1, \dots, \tilde{L}_B$ by a recursive method of matching sample cumulants with theoretical cumulants.

4. Finally we can calculate corrected percentiles $\tilde{F}^{-1}(1-\alpha/2)$ for $\tilde{S}_b = \tilde{L}_b + \hat{M}_b$. Consequently our improved bootstrap confidence interval for T is of the form

$$\{\hat{T}(Y) - \hat{\text{se}}_{\text{BOOT}} \tilde{F}^{-1}(\alpha/2), \hat{T}(Y) + \hat{\text{se}}_{\text{BOOT}} \tilde{F}^{-1}(1-\alpha/2)\}.$$

5. To obtain a bootstrap estimate V_b of

$$\text{var}\{\hat{T}(Y) - T(Y)\} = \text{var}\left\{\sum_{i \in S} (w_i - 1) \epsilon_i\right\} + \text{var}\left(\sum_{i \in S} \epsilon_i\right),$$

let $V_B = V_1 + V_2$ where V_1 is a bootstrap estimate for the first term

$$\text{var}\left\{\sum_{i \in S} (w_i - 1) \epsilon_i\right\} = \sigma^2 \sum_{i \in S} (w_i - 1)^2 G_i$$

and V_2 is an estimate of $\text{var}(\sum_{i \in S} \epsilon_i)$, perform the following steps:

- Bootstrap $\sum_{i \in S} \epsilon_i$ to obtain

$$\hat{\sigma}^2 \sum_{i \in S} G_i = \widehat{\text{var}}_{\text{BOOT}}\left(\sum_{i \in S} \epsilon_i\right);$$

- Assume

$$\sum_{i \in S} G_i / \sum_{i \in S} G_i \approx \sum_{i \in S} \Omega_i / \sum_{i \in S} \Omega_i$$

then

$$V_2 \approx \frac{\sum_{i \in S} \Omega_i}{\sum_{i \in S} \Omega_i} \widehat{\text{var}}_{\text{BOOT}}\left(\sum_{i \in S} \epsilon_i\right).$$

4. Simulation Study

4.1 Description of data

Our primary aim in this section is to see how the techniques developed in the previous section perform when applied to real survey data. In particular, we will see how the bootstrap variance estimator and confidence intervals compare to other more conventional techniques both when the assumed super-population model is correct and when it is misspecified.

Data for this analysis was obtained from the 1989 Agricultural and Grazing Industries Survey (AAGIS) run

by the Australian Bureau of Agriculture and Resource Economics. Three different regions were selected covering a wide range of climatic conditions and enterprise mixtures typically encountered in Australian broadacre agriculture.

4.2 Simulation results

In this section we summarize the results of a simulation study which compares the performance of the two bootstrap confidence intervals with more conventional confidence intervals.

To test the accuracy of the various confidence intervals against variance misspecification in model (1) we perform a simulation study based on the data introduced in section 4.1. The simulation is not the traditional 'repeated sampling' study, but is rather one in which the data is reconstructed to follow a specific model form while maintaining the structure of the original population.

In our tables the simple and the robust Dunstan-Chambers model-based methods are abbreviated to "Simple M-B" and "Robust D-C" respectively. Table 1 reports 95% confidence intervals calculated by the three different methods for each of the three different regions: A, B, and C. It can be seen that the simple model-based method gives the widest intervals. The bootstrap intervals grow narrower and stabilise as B increases. Note that here we are using Efron's efficient computation method for confidence intervals, otherwise we would require B to be much larger to attain similar results.

Table 2 reports the coverage probabilities for 95% confidence intervals based on the three methods. These coverage probabilities were estimated using $M = 5000$ simulations. When $\gamma = 1.0$ it can be seen that the Dunstan-Chambers robust model-based confidence interval performs best of all, the bootstrap confidence interval is slightly worse and the simple model-based method is very conservative. However, as γ moves away from 1.0, the bootstrap confidence interval remains fairly accurate and definitely superior to the Dunstan-Chambers robust model-based confidence interval, which has poorer coverage probability the further γ is from 1.0, while the simple model-based confidence interval remains overly conservative.

5. Conclusions

The above results indicate that both the bootstrap and robust Dunstan-Chambers model-based confidence intervals are considerably more reliable than the conventional simple model-based confidence interval, particularly under variance misspecification of the assumed model (1). Although the Dunstan-Chambers method is asymptotically equivalent to the bootstrap method, our simulation results reveal that the bootstrap method is still superior under more extreme variance misspecification. A heuristic explanation to this fact is that the bootstrap

method is better at tracking skewness and other anomalies in the sample distribution; a more rigorous investigation in this direction is a goal of our future research. Overall, by sacrificing some additional theoretical computation and programming efforts, a careful application of Efron's efficient bootstrap computation will help us to achieve reliable estimates especially when necessitated by financial and manpower constraints.

References

- Australian Bureau of Agricultural and Resource Economics (1991). Farm Surveys Report 1991, Australian Government Publishing Service, Canberra.
- Australian Bureau of Statistics (1983). Australian Standard Industrial Classification 1983 (Cat. No. 1204.0).
- Bardsley, P. and Chambers, R.L. (1984). Multivariate estimation from unbalanced samples. *J. Roy. Statist. Soc. Ser. B* **33**, 290-299.
- Chambers, R.L. (1986). Outlier robust finite population estimation. *J. Amer. Statist. Assoc.*, **81**, 1063-1069.
- Chambers, R.L. and Dunstan, R. (1986). Estimating distribution functions from survey data. *Biometrika*, **73**, 597-604.
- Do, K.-A. (1992). A simulation study of balanced and antithetic bootstrap resampling methods. *J. Statist. Comput. Simul.* (to appear).
- Do, K.-A. and Hall, P. (1992). Distribution estimation using concomitants of order statistics, with application to Monte Carlo simulation for the bootstrap. *J. Roy. Statist. Soc. Ser. B* **54**, (to appear).
- Dunstan, R. and Chambers, R.L. (1986). Model-based confidence intervals in multipurpose surveys. *J. Roy. Statist. Soc. Ser. C* **35**, 276-280.
- Efron, B. (1982). *The jackknife: the bootstrap, and other resampling plans*. CMBS 38 SIAM-NSF.
- Efron, B. (1990). More efficient bootstrap computations. *J. Amer. Statist. Assoc.* **85**, 79-89.
- Rao, J.N.K. and Wu, C.F.J. (1988). Resampling inference with complex survey data. *J. Amer. Statist. Assoc.* **83**, 231-241.
- Royall, R.M. and Cumberland, W.G. (1978). Variance estimation in finite population sampling. *J. Amer. Statist. Assoc.* **73**, 351-358.
- Royall, R.M. and Herson, J. (1973a). Robust estimation in finite populations, I. *J. Amer. Statist. Assoc.* **68**, 880-889.
- Royall, R.M. and Herson, J. (1973b). Robust estimation in finite populations, II. *J. Amer. Statist. Assoc.* **68**, 890-893.
- Silverman, B. (1986). *Density estimation*. Chapman and Hall, London.

Table 1. Approximate 95% confidence intervals for the weighted total of farm cash operating surplus

Region	Method	95% confidence intervals
A $\hat{T} = 3.55787 \times 10^8$	Simple M-B	$[2.69158 \times 10^8, 4.44415 \times 10^8]$
	Robust D-C ($\gamma = 1$)	$[2.92161 \times 10^8, 4.21412 \times 10^8]$
	Bootstrap	$B = 200$ $[2.93170 \times 10^8, 4.20402 \times 10^8]$
		$B = 500$ $[2.94063 \times 10^8, 4.28025 \times 10^8]$
B $\hat{T} = 2.93165 \times 10^8$	Simple M-B	$[2.28711 \times 10^8, 3.51614 \times 10^8]$
	Robust D-C ($\gamma = 1$)	$[2.40535 \times 10^8, 3.45786 \times 10^8]$
	Bootstrap	$B = 200$ $[2.42934 \times 10^8, 3.47387 \times 10^8]$
		$B = 500$ $[2.43003 \times 10^8, 3.45315 \times 10^8]$
C $\hat{T} = 1.28297 \times 10^8$	Simple M-B	$[8.37196 \times 10^7, 1.72874 \times 10^8]$
	Robust D-C ($\gamma = 1$)	$[1.02915 \times 10^8, 1.53679 \times 10^8]$
	Bootstrap	$B = 200$ $[1.03275 \times 10^8, 1.53379 \times 10^8]$
		$B = 500$ $[1.03264 \times 10^8, 1.53333 \times 10^8]$

Table 2. Monte Carlo estimates of coverage probabilities for 95% confidence intervals

Method	Region 431	Region 22	Region 122
Bootstrap	.955	.948	.949
Simple M-B	.977	.968	.972
Robust D-C $\gamma = 0$.960	.940	.963
Bootstrap	.949	.948	.952
Simple M-B	.972	.981	.966
Robust D-C $\gamma = 0.5$.954	.958	.960
Bootstrap	.943	.952	.949
Simple M-B	.965	.957	.978
Robust D-C $\gamma = 1.0$.951	.950	.954
Bootstrap	.950	.955	.953
Simple M-B	.965	.968	.970
Robust D-C $\gamma = 1.5$.962	.963	.964
Bootstrap	.960	.959	.957
Simple M-B	.978	.977	.980
Robust D-C $\gamma = 2.0$.963	.965	.969

Graphic Representation of the Effects of Skewness and Kurtosis on the Power of the Two-Sided Student's t Test

Mark Eakin, P.O. Box 19437, Information Systems Dept., University of Texas at Arlington, Arlington, Texas 76019

Sam Bowman, Alcon Laboratories, 6302 South Freeway,
Fort Worth, Texas 76133

Abstract

It is well known that skewness affects the power of student's t test more than does kurtosis. However, it is often difficult to estimate the relative importance of each of these factors when sampling from populations with different skewness and kurtosis.

This investigation evaluates the ability of graphic interpretation of the contributions of skewness and kurtosis to provide an understanding of the effect of each on the power of the one sample, two sided t test. Previous studies have used tables and mathematical interpretation to depict these effects. This study uses simulation to develop the power curves of size 5, 10, and 30. The Tadikamalla and Johnson system was used. to generate populations with different values of skewness and kurtosis. on both the distribution and power functions. Graphs of these functions are superimposed on a graph of skewness versus kurtosis.

The study verifies the effects of skewness and kurtosis in small samples taken from nonnormal populations. Graphic depiction emphasizes the importance of knowing the effects of skewness and kurtosis on the power of the t test for small sample sizes. The graphic presentation of the effects of skewness and kurtosis on both the distribution and power functions allows quick assessment of any potential deviation from the assumption of normality.

Introduction

Students are told in many textbooks that skewness affects the power of the Student's t-test more than kurtosis. However it is often unclear to the student the relative importance of each. If a one unit increase in skewness causes a certain effect on the power, then how much does kurtosis have to change in order to get the same effect on power? This paper examines graphically the effect of changes in skewness and kurtosis on the power curves of the one sample t-test. The power curves are generated by simulating and sampling from the Tadikamalla-Johnson family of distributions.

The effect of non-normal distributions on the t-test have been examined by Srivastava (1958) and Tiku (1971). Srivastava investigated the amount of non-normality that could be allowed in a near normal population without seriously affecting the significance level or the power of the Student's t-test and what the effects would be in different non-normal situations. His data demonstrated that the effect of skewness was greater on the probability of Type I error than that of kurtosis. The effect of skewness was shown to be more prominent when the kurtosis in the parent population was of low order, however, as the parent population increases in kurtosis (becomes highly leptokurtotic or platykurtotic), the effects of skewness and kurtosis on the power of the t test tend to negate each other. He cites the example of a parent population which is leptokurtotic and positively skewed. In this case, the power of the one sided t-test is close to the "normal theory" value. Finally, he points that the effect of increasing the sample size causes a decrease in the effect of non-normality on the power of the t-test.

Another investigation of the effect of non-normality of Student's t was done by Tiku (1971). His work showed clearly that, the power of the two sided t-test is not greatly affected and could be employed for a large number of symmetric, near symmetric, and moderately non-normal populations. It was also shown that, for moderately large sample sizes ($n \geq 30$), the effect of non-normality is negligible and the effect of non-normality on the Type I error is negligible even for small samples.

The approaches taken by Tiku and Srivastava differ markedly. Tiku derived an expression for the power of the two-sided t test by using the joint distribution of the mean and standard deviation of a sample. His approach concludes with the development of non-normality correction functions. The values of these functions are compared to evaluate the effect of Student's t under non-normal conditions on the power of the expression. Srivastava investigated the power of the one-sided t test using the first four terms of an Edgeworth series.

Both Tiku and Srivastava only reported a comprehensive table of power values for a narrow grid of skewness and kurtosis values and then for just a single sample size. For 20 degrees of freedom, Tiku gave power values of the one-sided t test for skewness of -.5, 0, .5,

and 1, kurtosis values of 2, 3, 4, 5, and 6, and noncentrality of 0, 1, ..., 4. For samples of size 10, Srivastava explored a grid of values for skewness of -.6, -.4, ..., .6, kurtosis of 2, 3, 4 and 5, and noncentrality of 0, 1, ..., 4 standard errors. However, the Srivastava power table, reported these values for a *t* test that has been adjusted for non-normality.

Both of the above approaches have deficiencies. They give complete power values for only a small number of noncentrality values and only for one sample size. Additionally, Srivastava reports values for an adjusted *t*-value. Furthermore, these approaches rely on the mathematical interpretation of the derived expressions by the reader to visualize the effect of parameters such as sample size and power. This paper graphically presents the power curves of an unadjusted, two-sided *t* test over a wide range of skewness and kurtosis values for different sample sizes and noncentrality.

Methodology

Construction of the power curves used a Monte Carlo simulation sampling from the Tadikamalla-Johnson (1979) Lu and Lb family of distributions. The Tadikamalla-Johnson family of distributions covers a region of skewness and kurtosis that is shown in Figure 1. Twenty-seven different combinations of skewness and kurtosis were chosen for this simulation. The probability density functions of these distributions are shown in Figure 1.

The simulation design contained three factors: sample size, skewness and kurtosis combinations, and noncentrality. The sample sizes consisted of $n=5, 10$, and 30. As mentioned, there were 27 different combinations of skewness and kurtosis. The null hypothesis stated that the mean of the population was zero against an alternative of not being zero. The true value of the mean was allowed to vary from 2 standard deviations below to 2 standard deviations above zero in increments of 0.125. Each combination of the three factors were replicated at least 100,000 times to obtain a value of power; some combinations were sampled up to 180,000 if necessary to achieve stable estimates of power. Figure 2 depicts the resulting power curves.

Conclusions

The power curves shown in Figure 2 graphically validate the results of Srivastava and Tiku in that the effects of skewness and kurtosis are not uniform. If one selects a value of skewness, of say 1.5, and then increases skewness, the lower curve approaches a more symmetric shape. These results agree with the observation

of Srivastava that not only is the effect of skewness is greater on the probability of Type I error than that of kurtosis but that there exists combinations where high kurtosis tends to reduce the effect of skewness.

Sample size has been well known to affect power. Sample sizes as large as 30 have power curves as symmetric as a normal distribution for any of the distributions depicted. However, sample sizes of 5 from populations having large skewness may have very deformed power curves with true Type I error rates as high as 0.3 for *t*-tests.

This paper extends the work of Tiku and Srivastava to a wider range of skewness, kurtosis, sample size, and noncentrality, illustrated possible density functions giving these skewness and kurtosis values, and depicted all the power curves simultaneously. The distributions in Figure 1 show the types of probability density functions that can generate certain combinations of skewness and kurtosis. The graphs presented in Figure 2 allow an investigator to easily determine the extent that non-normality in a "near-normal" population affects the significance level or the power of the Student's *t* test. The graphs also allow an investigator to quickly see how changing either skewness or kurtosis affects the power.

BIBLIOGRAPHY

- Srivastava, A. B. L., (1958), "Effect of nonnormality on the power function of *t*-test," *Biometrika*, 45, 421-429.
- Tadikamalla, P. R. and Johnson, N.L. (1982), "Systems of frequency curves generated by transformations of logistic variables", *Biometrika*, 69, 461-465.
- Tiku, M.L., (1971), "Student's *t* distribution under non-normal situations", *Australian Journal of Statistics*, 13 (3), 142-148.

Figure 1: Distribution Shapes for Simulation Values

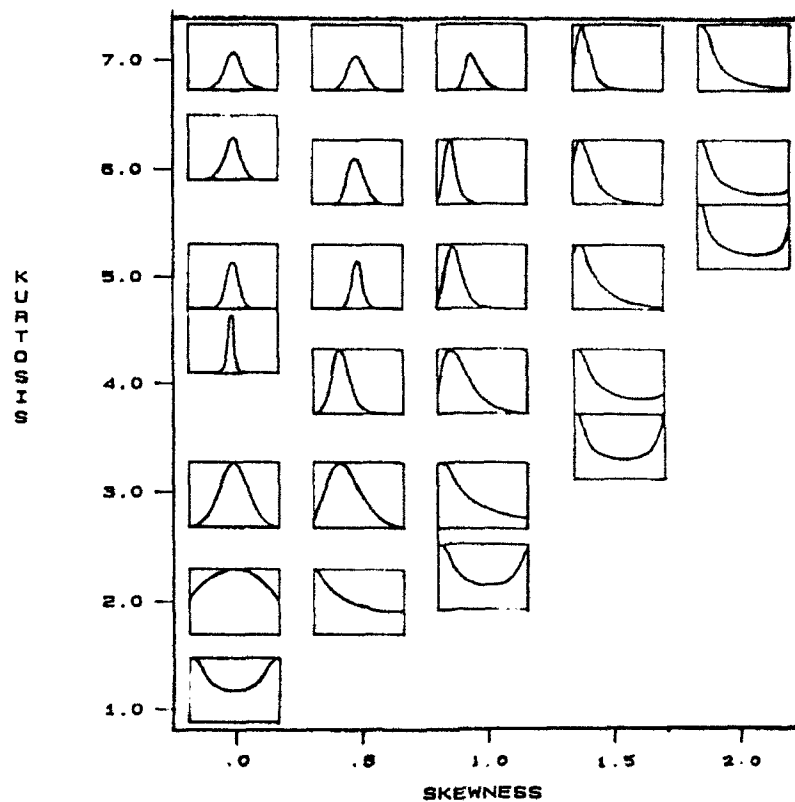
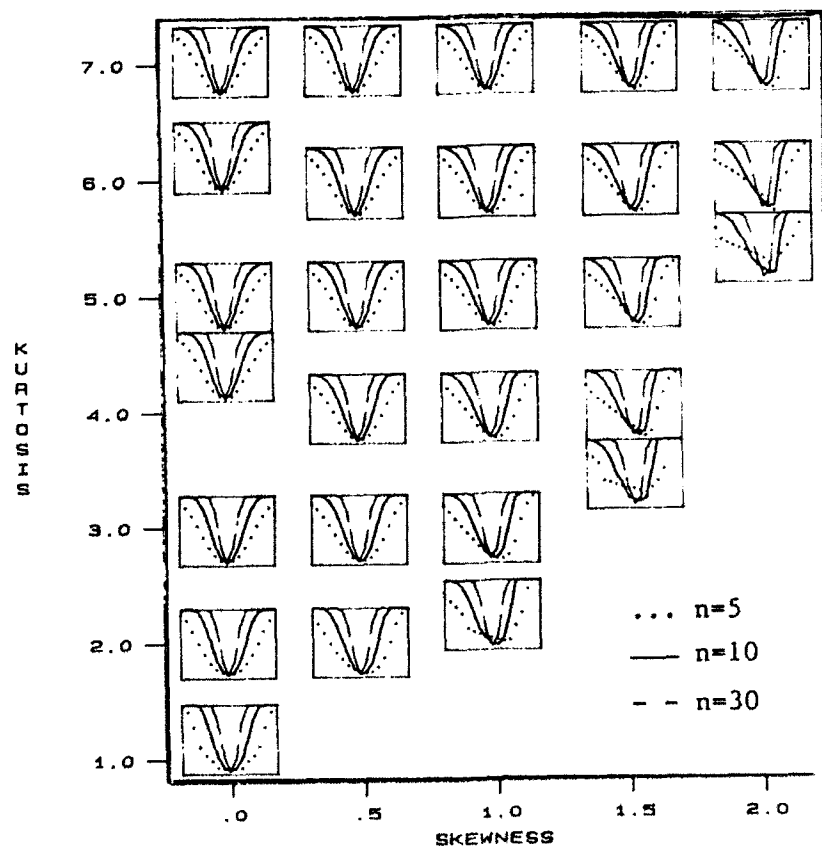


Figure 2: Power Curves for each Simulation Value



POWER COMPARISONS FOR TWO-SAMPLE RANK TESTS

Sudha Jain and J. G. C. Templeton
 Department of Industrial Engineering
 University of Toronto, Toronto
 Canada M5S 1A4

ABSTRACT

The asymptotic power of the Wilcoxon, Savage, Normal Score and Maximum Efficiency Robust tests is compared numerically for the two-sample scale problem when the underlying distribution is a Gamma distribution. In addition, the Maximum Efficiency Robust Test is shown preferable in comparison to Wilcoxon, Savage and Normal Score tests.

1. INTRODUCTION

Gastwirth and Mahmoud (1986) proposed a so-called Maximum Efficiency Robust Test (MERT) for the hypothesis that two samples come from a common distribution against the alternative that they differ in scale. They compared the asymptotic relative efficiency (ARE) of the MERT with the ARE's of the Savage and Normal Score tests for observations from a Gamma distribution. Woinsky (1972 b) and Whiteside, Duran and Boullion (1975) compared the ARE's of the Wilcoxon, Savage and the Sum of Squared Ranks tests for the two-sample scale problem using samples drawn from Gamma distributions. These tests can be used for comparing the performance of different queueing systems, for instance, the service time distributions of two different servers can be compared to see whether they have the same scale parameters. The performance comparison of the rank-tests required the estimation of the power of these tests. The asymptotic power of these tests will be developed based on Hajek and Sidak's (1967) work.

Section 2 deals with the developments of the linear rank-test statistics. The asymptotic power of the test statistics is defined based on the fundamental principle of Hajek and Sidak (1967). The asymptotic

relative efficiencies (ARE) of the linear rank tests are discussed in section 3.

Finally, in section 4, the power of the Wilcoxon, Savage, Normal Score and Maximum Efficiency Robust tests is computed when the underlying distribution belongs to the Gamma family for various values of the shape parameter. The Maximum Efficiency Robust test is recommended as the best applicable rank test for queueing models.

2. RANK-TESTS

Let X_1, X_2, \dots, X_N be independent random variables with continuous distribution functions $F_1(x), F_2(x), \dots, F_N(x)$ respectively. A linear rank statistic S is defined by

$$S = \sum_{i=1}^N c_i a(R_i), \quad (2.1)$$

where c_i 's are known regression constants, R_i is the rank of X_i among X_1, X_2, \dots, X_N and $a(R_i)$ is the value of the "score function" $a(\cdot)$ at R_i .

THE TWO-SAMPLE CASE

Let x_1, x_2, \dots, x_m be m independent observations on a random variable X with CDF $F(x)$ and $x_{m+1}, x_{m+2}, \dots, x_{m+n}$ be n independent observations on a random variable Y with CDF $G(y)$. Consider testing the hypothesis

$$H_0: p(x_1, x_2, \dots, x_N) = \prod_{i=1}^N f(x_i)$$

against an alternative differing in scale

$$H_1: q(x_1, x_2, \dots, x_N) = \exp\left(-\sum_{i=1}^N d_i\right)$$

$$\begin{aligned} & \times \left(\prod_{i=1}^m f(x_i \exp(-d_i)) \right) \\ & \times \left(\prod_{i=m+1}^N f(x_i) \right), \end{aligned} \quad (2.2)$$

where $d = (d_1, d_2, \dots, d_N)$ is an arbitrary vector and $N = m + n$.

Hajek and Sidak (1967) defined the following statistic S_c for testing the hypothesis (2.2):

$$S_c = \sum_{i=1}^N (c_i - \bar{c}) a(R_i), \quad (2.3)$$

$$\text{where } \bar{c} = \frac{1}{N} \sum_{i=1}^N c_i.$$

The asymptotic power of the one- and two-sided S_c tests at level α (Hajek and Sidak (1967)) is given by

$$\text{Power} = 1 - \Phi(Z_{1-\alpha} - \rho_1 \rho_2 b) \quad (2.4)$$

for the one-sided test and

$$\text{Power} = 1 - \Phi(Z_{1-\alpha/2} - \rho_1 \rho_2 b) + \Phi(Z_{\alpha/2} - \rho_1 \rho_2 b) \quad (2.5)$$

for the two-sided test, where notations b , ρ_1 and ρ_2 have the usual meanings given by Hajek and Sidak (1967, page 268), and $\Phi(x)$ is the standard normal cumulative distribution function (cdf).

REMARK. Let $c_1 = c_2 = \dots = c_m = 1$; $c_{m+1} = c_{m+2} = \dots = c_N = 0$;

$d_1 = d_2 = \dots = d_m = \Delta$; $d_{m+1} = d_{m+2} = \dots = d_N = 0$; and $c_i = d_i = 0$ otherwise. Therefore,

$$\bar{c} = m/N, \text{ and } \bar{d} = m\Delta/N.$$

It can be easily noted that

$$\rho_2 = 1, \text{ for } \Delta > 0, \quad (2.6)$$

$$\rho_2 = -1, \text{ for } \Delta < 0 \quad (2.7)$$

and

$$b^2 = I_1(f) \Delta^2 (mn/N), \quad (2.8)$$

where

$$I_1(f) = \int_{-\infty}^{\infty} (-1-x) \frac{f'(x)}{f(x)} f(x) dx. \quad (2.9)$$

3. ASYMPTOTIC RELATIVE EFFICIENCY OF RANK TESTS

Hajek and Sidak (1967, page 268) proved that the asymptotic relative efficiency of the S_c -test when

testing H_0 against the alternative (2.2) is

$$ARE = e = \rho_1^2 \rho_2^2. \quad (3.1)$$

From (3.1) and (2.4) the asymptotic power of the S_c -test for one-sided alternatives equals

$$\text{Power} = 1 - \Phi(Z_{1-\alpha} - e^{1/2} b). \quad (3.2)$$

Similarly, the asymptotic power of the S_c -test for two-sided alternatives equals

$$\text{Power} = 1 - \Phi(Z_{1-\alpha/2} - e^{1/2} b) + \Phi(Z_{\alpha/2} - e^{1/2} b). \quad (3.3)$$

WILCOXON TEST

If we put $a(i) = i$ and $c_i = 1$ for $1 \leq i \leq m$ and $c_i = 0$ for $m < i \leq N$ in linear rank statistics (2.1), we obtain the two-sample Wilcoxon statistic

$$S = \sum_{i=1}^m R_i. \quad (3.4)$$

For the Gamma density the general form of the ARE for Wilcoxon statistic is given by Woinsky (1972 b) as follows:

$$ARE_{(Wilcoxon)} = \frac{12}{k} \left[\frac{\Gamma(2k)}{2^{2k} \Gamma^2(k)} \right]^2, \quad (3.5)$$

where k is the shape parameter.

Applying Stirling's approximation to the Gamma function to (3.5), we have

$$ARE_{(Wilcoxon)} = \frac{3}{\pi} \exp(-1/4k) + O(k^{-3}). \quad (3.6)$$

Thus, as $k \rightarrow \infty$, the Gamma density approaches normal density and the ARE is given by

$$\lim_{k \rightarrow \infty} ARE_{(Wilcoxon)} = \frac{3}{\pi}. \quad (3.7)$$

VAN DER WAERDEN (or NORMAL SCORE) TEST

The Van der Waerden statistic (or Normal Score statistic) is determined by

$$S = \sum_{i=1}^m a(R_i), \quad (3.8)$$

where

$$a(R_i) = E(S_{(i)})$$

and $S_{(1)} < \dots < S_{(m+n)}$ are the $m+n$ order statistics from the $N(0,1)$ distribution.

SAVAGE TEST

The Savage statistic is defined by

$$S = \sum_{i=1}^N a(R_i), \quad (3.9)$$

where

$$a(i) = \sum_{j=N-i+1}^N \left(\frac{1}{j}\right).$$

It can be shown that

$$\sum_{j=N-i+1}^N \frac{1}{j} = E[-\ln(1-U^{(i)})], \quad (3.10)$$

where $U^{(i)}$ is the i^{th} order statistic in a sample of size N from the uniform distribution over $(0,1)$.

MAXIMUM EFFICIENCY ROBUST TEST

Gastwirth and Mahmoud (1986) proposed a Maximum Efficiency Robust Test (MERT) based on a rank test as follows:

$$R(u) = \frac{[-\ln(1-u)-1+\Phi^{-1}(u)]}{(2(1+\rho))^{1/2}}, \quad (3.11)$$

where

$$\rho = \int_0^1 (-\ln(1-u)-1)\Phi^{-1}(u)du.$$

Define Z_1 and Z_∞ by

$$Z_1 = -\ln(1-u)-1 \quad (\text{Savage component})$$

and

$$Z_\infty = \Phi^{-1}(u) \quad (\text{Normal Score component}).$$

Thus, the standard normal form of MERT (3.11) is given by

$$R(u) = \frac{Z_1 + Z_\infty}{(2(1+\rho))^{1/2}}. \quad (3.12)$$

The value of ρ was obtained by numerical integration, and is approximately equal to 0.90228. Hence, the MERT test is given by

$$R(u) = \frac{Z_1 + Z_\infty}{1.9505}. \quad (3.13)$$

The ARE of the Normal Score, Savage and MERT when the underlying distribution is Gamma given by Gastwirth and Mahmoud (1986).

4. APPLICATION

To show how the rank-tests would work in queueing models, we will illustrate their application to Gamma distribution for various values of the shape

parameter. The power of the various rank-tests discussed in section 3 will be compared.

For the Gamma density function

$$f(x) = \frac{\lambda^k x^{k-1} \exp(-\lambda x)}{\Gamma(k)}, \quad x > 0, k, \lambda > 0,$$

we have from (2.9)

$$I_1(f) = \int_0^\infty (\lambda x - k)^2 \frac{\lambda^k x^{k-1} \exp(-\lambda x)}{\Gamma(k)} dx. \quad (4.1)$$

The integral (4.1) can be easily evaluated and is given by

$$I_1(f) = k. \quad (4.2)$$

Hence, b^2 for the Gamma distribution is given by

$$b^2 = k \Delta^2 \frac{mn}{m+n}, \quad (4.3)$$

where $\Delta = \ln\left(\frac{\lambda_1}{\lambda_2}\right)$.

To apply the theoretical formulas developed so far, power of the various tests was estimated for testing the equality of scale parameter for Gamma distribution, assuming that the shape parameters are equal.

If the shape parameter k equals one, then the Gamma distribution reduces to the exponential distribution, that is

$$f(x) = \lambda \exp(-\lambda x). \quad (4.4)$$

Comparisons of Power for various rank-tests are presented in tables 4.1 and 4.2 for various sample sizes when the underlying distribution is exponential.

Tables 4.3 and 4.4 present the power of the rank-tests for Gamma distribution with shape parameter $k=2$ and for different scale parameters. Table 4.5 presents the power when the shape parameter $k=4$ and the scale parameters are $\lambda_1=1.0$ and $\lambda_2=2.0$.

5. CONCLUDING REMARKS

Tables 4.1 to 4.5 revealed that for a given shape parameter k , all the four tests are more powerful with the increase in the number of observations. The Savage test is more powerful than the other three rank-tests for the exponential distribution. It is well-known that the Savage statistic is the optimum rank statistic for an exponential distribution (Savage, 1956). The Wilcoxon is the least powerful test for the value of scale parameter considered for any fixed value k ; however, as the value of shape parameter k increases all four rank-tests become consistently more

powerful for a fixed value of the scale parameters. The tables 4.3, 4.4 and 4.5 showed that the power of the MERT is maximum in comparison to the Wilcoxon, Savage, Normal Score tests when the shape parameter for Gamma distribution is equal to 2 and 4 respectively. The power of all the four rank-tests approaches maximum when the difference between the two sample sizes decreases to zero. Further, MERT has the advantage in terms of its simplicity to use and is also more powerful than other rank-tests.

REFERENCES

- Gastwirth, J.L. and Mahmoud, H. (1986). An efficiency robust nonparametric test for scale change for data from a Gamma distribution. *Technometrics*, Vol. 28, 1, 81-84.
- Gastwirth, J.L. (1970). On robust rank test. In *Nonparametric Techniques in Statistical Inference*, ed. M. Puri, 89-109. Cambridge Univ. Press.
- Gibbons, J.D. (1964). On the power of two-sample rank-tests on the equality of two distribution functions. *J. Royal Statist. Soc. Ser. B*, 26, 293-304.
- Gross, D. (1975). Sensitivity of queueing models to the assumption of exponentiality. *Naval Res. Log. Quart.* Vol. 22, 271-287.
- Hajek, J. and Sidak, Z. (1967). *Theory of rank-tests*. Academic Press.
- Hajek, J. (1969). *A course in nonparametric statistics*. Holden-Day.
- Hettmansperger, T.P., (1984). *Statistical inference based on ranks*. Wiley.
- Klotz, J. (1962). Nonparametric tests for scale. *Annals of Math. Stat.* 33, 498-512.
- Savage, I.R. (1956). Contributions to the theory of rank order statistics--the two sample case. *Annals of Math. Stat.* Vol. 27, 590-615.
- Shiu, W.K. and Bain, L.J. (1983). A two-sample test of equal Gamma distribution scale parameters with unknown common shape parameter. *Technometrics*, Vol. 25, 4, 377-381.
- Whiteside, B.S., Duran, B.S. and Boullion, T.L. (1975). A comparison of some nonparametric tests for scale. *J. Statist. Comput. Simul.* Vol. 4, 121-132.
- Woinsky, M.N. (1972 a). Nonparametric detection using spectral data. *IEEE Trans. on Infor. Theory*. 18, 120-126.
- Woinsky, M.N. (1972 b). A composite nonparametric test for a scale slippage alternative. *Annals of Math. Stat.*, 43, 65-73.

TABLE 4.1. Power of the Wilcoxon, Savage, Normal Score and MERT for the exponential distribution with $\lambda_1 = 1.0$ and $\lambda_2 = 2.0$ at significance level $\alpha = 0.05$.

m	n	b	Power			
			Wilcoxon	Savage	Normal Score	MERT
5	15	1.3423	0.3147	0.3805	0.3322	0.3682
8	12	1.5187	0.3708	0.4491	0.3918	0.4346
10	10	1.5500	0.3811	0.4615	0.4027	0.4466
10	20	1.7898	0.4621	0.5567	0.4880	0.5396
12	18	1.8600	0.4863	0.5843	0.5133	0.5667
15	15	1.8984	0.4996	0.5992	0.5271	0.5814
15	25	2.1224	0.5765	0.6826	0.6064	0.6641
18	22	2.1810	0.5963	0.7032	0.6266	0.6847
20	20	2.1920	0.5999	0.7069	0.6304	0.6885
20	40	2.5311	0.7078	0.8115	0.7385	0.7945
25	35	2.6471	0.7413	0.8411	0.7714	0.8251
30	30	2.6847	0.7517	0.8501	0.7815	0.8344
30	50	3.0016	0.8301	0.9120	0.8562	0.8998
35	45	3.0757	0.8458	0.9232	0.8708	0.9119
40	40	3.1000	0.8507	0.9267	0.8754	0.9157
45	55	3.4485	0.9101	0.9640	0.9288	0.9569
50	50	3.4659	0.9125	0.9654	0.9309	0.9585
60	100	4.2448	0.9789	0.9953	0.9856	0.9937
80	80	4.3841	0.9843	0.9969	0.9896	0.9957
90	110	4.8770	0.9950	0.9993	0.9971	0.9990
100	100	4.9015	0.9953	0.9994	0.9973	0.9991

TABLE 4.2. Power of the Wilcoxon, Savage, Normal Score and MERT for the exponential distribution with $\lambda_1 = 1.0$ and $\lambda_2 = 3.5$ at significance level $\alpha = 0.05$.

m	n	b	Power			
			Wilcoxon	Savage	Normal Score	MERT
12	18	3.3615	0.8972	0.9566	0.9174	0.9486
15	15	3.4308	0.9076	0.9626	0.9266	0.9553
15	25	3.8358	0.9532	0.9856	0.9653	0.9819
18	22	3.9417	0.9615	0.9891	0.9720	0.9859
20	20	3.9616	0.9629	0.9896	0.9732	0.9867
25	35	4.7840	0.9937	0.9991	0.9962	0.9987
30	30	4.8519	0.9947	0.9993	0.9969	0.9989

TABLE 4.3. Power of the Wilcoxon, Savage, Normal Score and MERT for the Gamma distribution ($k=2$) with $\lambda_1 = 1.0$ and $\lambda_2 = 2.0$ at significance level $\alpha = 0.05$.

m	n	b	Power			
			Wilcoxon	Savage	Normal Score	MERT
15	25	3.0014	0.8669	0.9077	0.8842	0.9091
18	22	3.0843	0.8826	0.9205	0.8989	0.9218
20	20	3.0998	0.8854	0.9228	0.9013	0.9240
20	40	3.5794	0.9498	0.9712	0.9593	0.9719
25	35	3.7434	0.9635	0.9804	0.9711	0.9809
30	30	3.7965	0.9673	0.9827	0.9743	0.9831
30	50	4.2446	0.9879	0.9947	0.9912	0.9949
35	45	4.3495	0.9906	0.9961	0.9932	0.9962
40	40	4.3838	0.9913	0.9965	0.9938	0.9966

TABLE 4.4. Power of the Wilcoxon, Savage, Normal Score and MERT for the Gamma distribution ($k=2$) with $\lambda_1 = 1.0$ and $\lambda_2 = 3.5$ at significance level $\alpha = 0.05$.

m	n	b	Power			
			Wilcoxon	Savage	Normal Score	MERT
5	9	3.1763	0.8984	0.9332	0.9132	0.9343
6	8	3.2805	0.9144	0.9455	0.9279	0.9465
7	7	3.3145	0.9192	0.9496	0.9322	0.9501
8	12	3.8815	0.9726	0.9860	0.9788	0.9864
10	10	3.9616	0.9769	0.9886	0.9823	0.9989

TABLE 4.5. Power of the Wilcoxon, Savage, Normal Score and MERT for the Gamma distribution ($k=4$) with $\lambda_1 = 1.0$, and $\lambda_2 = 2.0$ at significance level $\alpha = 0.05$.

m	n	b	Power			
			Wilcoxon	Savage	Normal Score	MERT
4	6	2.1476	0.6514	0.6711	0.6706	0.6906
5	5	2.1919	0.6668	0.6883	0.6860	0.7060
6	8	2.5669	0.7842	0.8041	0.8021	0.8202
7	7	2.5935	0.7915	0.8112	0.8092	0.8270
8	12	3.0372	0.8910	0.9057	0.9042	0.9170
10	10	3.0998	0.9017	0.9156	0.9142	0.9261
15	25	4.2446	0.9912	0.9936	0.9934	0.9952
20	20	4.3838	0.9939	0.9957	0.9955	0.9968

SMALL SAMPLE EMPIRICAL CRITICAL VALUES
AS A TOOL FOR THE COMPARISON OF
MULTIVARIATE NORMALITY GOF TESTS.

Jorge Luis Romeu
Mathematics
SUNY-Cortland, NY 13045
jromeu@suvvm.bitnet

Abstract.

Existing Multivariate Normality (MVN) Goodness of Fit (GOF) Tests either follow a known asymptotic distribution (e.g. Mardia's) or are empirical (e.g. Malkovich and Afifi). When samples are small the asymptotic theory cannot be safely invoked. Hence, their asymptotic distribution percentiles are no longer accurate. In such cases empirical critical values (ecv) are derived via Monte Carlo.

We have thus obtained ecv's for eight well known MVN GOF tests for $n=25(25)200$, $p=2(1)6(2)10$ and medium and high p -variate correlations. Using the ecv's we statistically study several characteristics of the unknown small sample distributions of these MVN tests. Then we present criteria as to when and where the asymptotic critical values can be safely used.

1.0 Introduction and Background.

This research stems from the work to demonstrate our newly developed MVN GOF test (Ozturk and Romeu, 1992). We conducted an extensive Monte Carlo power study (under a Cornell Theory Center award) to compare it with eight carefully selected and well established MVN GOF tests: Mardia's Skewness and Kurtosis, Royston's W, Cox and Small, Malkovich-Afifi, Hawkins and Koziol's Angles and Chi Square tests. For space and brevity we

refer the reader to Romeu (1990), for a complete list of references.

We soon realized how several of these MVN tests lacked any asymptotic distributions. And those who had one, converged to it very slowly, rendering them impractical when samples were "small". We also observed how some of these tests (i) were prone to detect certain types of departure from multivariate normality (say skewness) but not others (say kurtosis). Or how (ii) correlation among p -variates affected some tests very seriously. Or how the ecv's were (iii) severely affected by sample size or (iv) by number of p -variates. Or how (v) certain algorithm results varied from one computer to the other. Or a combination of any of the above mentioned problems.

We wanted to further investigate such problems and to study how the different tests fared under them. But, lacking the theoretical tool to undertake such comparative study (i.e. small sample distributions) we took an indirect approach. We thus, used the ecv's obtained by simulating five or ten thousand test results, to characterize the unknown statistical distributions.

2.0 Research Methods.

The validation of our Monte Carlo study is discussed elsewhere (Romeu, 1992). In this paper we present the statistical results regarding the mentioned problems of the eight MVN GOF tests.

First, the effect of sample size was investigated by regressing ecv's on inverse sample size $1/n$, for fixed number of p-variates and percentile (PCT). Some results, for bivariate normals, are shown in Table 1. Each test (MTD), their asymptotic critical values (CV), the regression independent term (Bo) and Index of Fit (IF) are given. A 95% confidence interval (CI) for each regression's Bo, for Mardia's Skewness (MSK) and Cox-Small (Cox) test, cover their asymptotic CV. However, Mardia's Kurtosis (MKT), converges much slower and its 95% CI for Bo doesn't cover its asymptotic CV.

Table 1: Regression $ecv = f(n)$.

MTD	PCT	Bo	CV	IF
MSK	0.90	7.79	7.78	0.99
MSK	0.95	9.67	9.49	0.98
MSK	0.99	13.77	13.28	0.98
MKT	0.90	1.63	1.65	0.99
MKT	0.95	2.08	1.95	0.99
MKT	0.99	2.90	2.58	0.92
COX	0.90	4.58	4.61	0.98
COX	0.95	5.99	5.99	0.94
COX	0.99	9.10	9.21	0.94

Next we investigated the "smallest" sample size n , required for the safe use of asymptotic critical values. Table 2 show examples of coverage (COV) of the 95th asymptotic percentile (CV) by a 95% non parametric CI, derived for that sample size (n) and no. of p-variates (p). Verify how, for different n and p , methods varied widely. Again, Mardia's Skewness ecv's cover the true asymptotic percentile for $n=200$ while Kurtosis, which converges at a slower rate, doesn't.

Table 2: Approximate CI for ecv's.

MTD	n	p	COV	n	p	COV
MSK	50	2	NO	50	8	NO
MSK	100	2	NO	100	8	NO
MSK	200	2	YES	200	8	YES
MKT	50	2	NO	50	8	NO
MKT	100	2	NO	100	8	NO
MKT	200	2	NO	200	8	NO

Next we investigated the effect of p-variate correlation. In practice, the covariance matrix is seldom known. Instead, it is estimated from the data and used in the GOF tests. In our power study we had also observed wide differences for low and high correlation.

We investigated this problem by taking, at prefixed and regular intervals of the order statistics (at $0.9(0.05)0.995$) differences of ecv's obtained for $\rho=0.5$ and 0.9 . There were two alternatives: (i) this difference was statistically zero (no correlation effect) or (ii) there was an effect of correlation. If so, this effect produced a shift problem or a scale one. Hence, the differences in ecv's would (or would not) be independent of how far out they were obtained. In the first case we used (i) paired (Wilcoxon/Sign) tests. In the second, (ii) regression of these differences on its percentiles. If there was a significance difference in (i) above, we obtained its 95% non parametric CI.

In Table 3 we report some of these non parametric 95% confidence intervals for ecv differences, for those methods that showed none or very small effect of p-variate correlation.

Table 3: CI for ecv Differences

MTD	p	Lower	Upper
Skewness	2	-0.22	-0.16
Skewness	4	0.12	0.26
Hawkins	2	0.018	0.025
Hawkins	4	-0.008	0.015
Cox-Small	2	-0.077	-0.012
Kurtosis	4	0.047	0.062

We also investigated the effect of the number of p-variates on ecv's. Several MVN methods (e.g. Koziol) required large n and small p for its use, which is not always met in practice. We again used regression of ecv's on p-variates (for p=2,3,4,5,6,8,10) for fixed sample size and asymptotic percentile. We verified how, with the exception of Koziol's Angles and Hawkins' tests, all others heavily depend on p-variates too.

We then reanalyzed ecv differences for low/high correlation, now for fixed asymptotic percentile (PC) and n, but increasing no. of p-variates. We verified how, with the exception of Royston's and Koziol's Angles tests, all other power results could be pooled. An example is shown in Table 4.

Table 4: 95% CI for ecv diff(p).

Test	PC	n	LB.	UB.
Skewness	0.90	25	-0.48	0.13
Skewness	0.95	25	-0.47	0.36
Skewness	0.99	25	-1.02	1.40

Of practical consideration is the effect of hardware on the calculation of ecv's. We observed some variation for two tests (Royston's and Koziol's Angles). We performed non

parametric paired comparisons between ecv's obtained in Syracuse University's IBM 3090 and Cornell's Supercomputer. Differences (fixed p=2, n=50) for correlations of 0.5 and 0.9 were obtained for successive ecv percentiles 0.9(0.005)0.995. We followed the same approach above described for the analysis of correlation effect. Descriptive statistics of some of our analyses are presented in Table 5. For relative comparison of the effect of these differences on ecv's, the mean ecv value, by method, is also given.

Table 5: Hardware effect comparison.

MTD	Q.25	Q.5	Q.75	Mean
MSK	-0.048	-0.020	0.075	8.75
ROY	0.883	0.975	1.208	5.26
HAW	0.008	0.014	0.039	1.30
CHI	0.002	0.005	0.006	0.22

Notice how ecv's for Royston's test vary in the order of 15%, while those for Skewness (MSK), Hawkins and Koziol's Chi Square tests are, for practical purposes, negligible. We conjecture that calculation of sensitive quantities (e.g. eigenvalues/eigenvectors) in the denominators of Koziol's Angles and Royston's algorithms account for such large differences, when processed in two significantly different machines as those used.

A complete set of tables of small sample ecv's for the MVN GOF tests discussed in this paper can be found in Romeu and Ozturk (1991).

3.0 Research Results.

The main result of this paper concerns the determination of the appropriate sample size for asymptotic values. Mardia's skewness test requires more than

100 observations before the use of asymptotic critical values is appropriate. The same holds for Cox and Small and Koziol. For $n=200$, our ecv's 95% CI results cover the asymptotic values. Mardia's Kurtosis test converges much slower and 95% CI obtained for $n=200$ do not cover the asymptotic percentiles. Hence, a sample of size 200 is still inadequate for using asymptotic critical values. Since, in practice, samples may be much smaller than that, our empirical critical values provide a useful tool for the practitioner.

Our ecv's regression results also indicate that test that don't have a known asymptotic distribution (e.g. Malkovich and Afifi) also converge as a function of $1/n$. Hence, a function may be found that approximates this test's unknown asymptotic distribution for large samples.

The next result of interest pertains to the effect of p-variate correlation on the power of the tests (or equivalently on their ecv's). This is of importance, since the covariance matrix is generally unknown and estimated from the samples. We concluded that only two procedures, Royston's W and Koziol's Angles test (as well as the Sigma Inverse implementation of our own multivariate Qn test) are seriously affected by p-variate correlation. Separate ecv's have been provided for medium (0.5) and high (0.9) rho, for those two tests. All other methods analyzed may be considered, for practical purpose, as approximately correlation free and single tables of ecv's are provided.

Two MVN GOF tests have been found

quite sensitive to hardware effect: Koziol's Angles and Royston's W. We caution the practitioner to calibrate our results with those of his own machine before using our ecv's.

Finally, we have also shown how empirical critical values, obtained from simulation, can become effective tools. The statistical study and comparison of the small sample (unknown) distributions of these MVN GOF tests was, both, required but infeasible with the conventional research tools (closed form distribution). We had few other alternatives, since the true distributions were either unknown or available only when the sample size was very large, which rendered them useless for our needs.

The use of the ecv's as a characterization of these unknown small sample distributions allowed us to investigate this problem.

4.0 Bibliography.

Ozturk A. and J. L. Romeu (1992). A New Method for Assessing Multivariate Normality With Graphical Applications. Comm. in Stat. -Simula. (21)1, 15-34.

Romeu, J. L. (1990). Development and Evaluation of a General Procedure for Assessing Multivariate Normality. CASE Center Tech. Report 9022. Syracuse University, Syracuse NY 13244.

Romeu, J. L. (1992) Small Sample Empirical Critical Values For Multivariate Normality Tests. ASA Winter Conference. Louisville, KY.

Romeu, J. L. and A. Ozturk (1991). A Comparative Study of Goodness of Fit Tests for Multivariate Normality. (Submitted for publication).

Table 6: Example of ecv's power comparison results (n=25; p=2).

PERCENT REJECTIONS FOR N= 20000 TOTAL CASES.			
METHOD:	ALPHA=0.10	ALPHA=0.05	ALPHA=0.01
CHOLESKI	0.09710	0.04675	0.00920
SIGMA	0.09755	0.04845	0.01025
M-SKEW	0.09860	0.04645	0.00910
M-KURT	0.09960	0.04975	0.01060
COX-SMAL	0.09560	0.04860	0.00895
ROYSTONW	0.10585	0.05415	0.01065
MALKOV	0.09960	0.04860	0.00910
KOZ-CHI	0.10155	0.05135	0.00985
KOZANGLE	0.10230	0.05140	0.00975
HAWKINS	0.10150	0.05100	0.01005

TABLE NO. 7		CRITICAL VALUES FOR THE CASE P = 2 VARIATES.									
RHO=0.5	SKEWNESS	KURTOSIS	ROYSTON	MALKOVICH	KOZIOL	COX-SMAL	HAWKINS	KOZIOL	CHI-SQR.	REG	TEST
N	%	LOWER	UPPER	N	%	N	%	N	%	N	%
25	90	5.87	-1.22	0.88	4.56	0.813	0.167	5.13	1.036	4.53	
25	95	7.48	-1.33	1.24	5.98	0.895	0.271	6.90	1.290	5.87	
25	99	11.24	-1.52	2.03	9.28	0.855	0.317	11.62	1.937	9.07	
50	90	6.67	-1.37	1.19	4.50	0.853	0.173	4.73	1.074	4.45	
50	95	8.27	-1.53	1.58	5.93	0.945	0.219	6.29	1.346	5.94	
50	99	12.18	-1.76	2.49	9.29	0.926	0.337	9.97	1.997	9.15	
75	90	7.03	-1.42	1.39	4.07	0.965	0.172	4.79	1.067	4.59	
75	95	9.06	-1.58	1.84	5.43	0.959	0.218	6.28	1.321	5.93	
75	99	13.07	-1.90	3.12	8.61	0.946	0.338	9.75	1.977	8.76	
100	90	7.33	-1.44	1.45	4.63	0.971	0.170	4.71	1.051	4.62	
100	95	9.14	-1.61	1.85	5.98	0.967	0.217	6.23	1.305	6.06	
100	99	13.26	-1.95	2.89	9.43	0.957	0.326	9.68	1.931	9.48	
125	90	7.35	-1.52	1.45	4.31	0.975	0.180	4.72	1.077	4.52	
125	95	9.35	-1.70	1.88	5.60	0.971	0.227	5.99	1.343	5.82	
125	99	13.35	-2.05	2.93	8.93	0.963	0.337	9.34	1.988	8.74	
150	90	7.44	-1.47	1.47	4.25	0.977	0.171	4.79	1.035	4.64	
150	95	9.48	-1.70	1.87	5.80	0.973	0.218	6.14	1.293	5.85	
150	99	13.61	-2.11	2.80	9.05	0.967	0.324	9.65	1.942	9.06	
175	90	7.62	-1.48	1.55	4.41	0.978	0.170	4.71	1.044	4.67	
175	95	9.40	-1.69	2.02	5.88	0.976	0.214	6.35	1.303	6.12	
175	99	13.07	-2.06	2.78	9.25	0.970	0.347	9.33	2.025	9.37	
200	90	7.64	-1.51	1.57	4.73	0.979	0.174	4.62	1.063	4.61	
200	95	9.43	-1.75	1.99	6.31	0.977	0.220	6.10	1.319	5.99	
200	99	13.37	-2.12	3.02	9.94	0.972	0.350	9.68	1.986	9.07	

On the Accuracy of Binomial and Proportion Estimators: an Absolute Rule

Eugene F. Schuster

Department of Mathematical Sciences
El Paso, Texas, 79968-0514

Abstract

We consider the problems of estimating: 1) a probability p , 2) a binomial parameter p and, 3) the proportion p of a finite population of size N having a given attribute. In each case, our estimator is the usual proportion of successes \hat{p} in a random sample of size r . Our main result is a simple, absolute rule: $\hat{p} \pm 1/\sqrt{r}$ is always at least a 91.0% confidence interval (C.I.) for the parameter p . In case 3), this rule holds whether sampling is with or without replacement. In fact, we have shown that $h(p) = h(p; r, N) = \text{Prob}(|\hat{p} - p| \leq 1/\sqrt{r})$ is at least as large under the hypergeometric model of simple random sampling without replacement as it is under the corresponding binomial model of random sampling with replacement. The significance of our rule is that it is a good, easily stated accuracy rule, holding for all r , N , and p , which can easily be understood by the layman when assessing accuracy of the estimator \hat{p} and discussing the relationship between accuracy and sample size. We give a table which indicates the lowest probability coverage of p by $\hat{p} \pm 1/\sqrt{r}$ for each $r \leq 200$. The lowest probability coverage is not monotone in r as one might expect. There are only 18 sample sizes where the probability coverage is below 0.94. In all three cases, the probability coverage rounds to at least 0.93 for $r \geq 13$, rounds to at least 0.940 for $r \geq 31$, and is less than the asymptotic normal value 0.9545 for $2 \leq r \leq 700$ in the binomial case.

1. Introduction and Summary

In Schuster (1978), we gave a good easily stated rule regarding the accuracy of probability estimates:

Theorem 1.1 *Let A be an event associated with a random experiment ξ . Suppose we make r independent trials of ξ and A occurs X times in these r trials. If we estimate $p = \text{Prob}(A)$ by $\hat{p} = X/r$ then at least 91.0% of the time (i.e., with probability at least 0.910) \hat{p} will be within $1/\sqrt{r}$ of p for any r and p .*

Noting that it is sufficient to consider $p \leq 0.5$, we proved Theorem 1.1 via three cases: Chebyshev's inequality easily handled $p \in [0, 0.1]$; published results on

the accuracy of the normal approximation to the binomial inferred the theorem for $p \in (0.1, 0.5]$ with samples sizes $r > 200$; finally, numerical methods were used to analyze the remaining possibilities, $0.1 < p \leq 0.5$ and $1 \leq r \leq 200$.

Of course, the same theorem holds when estimating a binomial proportion p by the proportion of successes \hat{p} which occur in r trials, i.e., $\hat{p} \pm 1/\sqrt{r}$ is always at least a 91.0% confidence interval (C.I.) for a binomial parameter p .

Several people, working with sample survey type data, have inquired if a corresponding theorem holds in the hypergeometric case of estimating the proportion p , having a given attribute (say success) in a finite population of N , by the proportion \hat{p} of successes in a simple random sample without replacement from the population. Our answer to this question is in the affirmative:

Theorem 1.2 *Let the random variable X record the number of successes in a random sample (with or without replacement) of size r , $1 \leq r \leq N$, from a population of N containing n successes. If we estimate $p = n/N$ by $\hat{p} = X/r$ then at least 91.0% of the time (i.e., with probability at least 0.910) \hat{p} will be within $1/\sqrt{r}$ of p for any r , N and p . Furthermore, $h(p) = h(p; r, N) = \text{Prob}(|\hat{p} - p| \leq 1/\sqrt{r})$ is at least as large under the hypergeometric model of simple random sampling without replacement as it is in the corresponding binomial model of random sampling with replacement.*

The details of the proof of Theorem 1.2, are given in Schuster (1992) where we prove Theorem 1.2 from Theorem 1.1, utilizing results following from Uhlmann's (1966) relations between the binomial and hypergeometric cumulative distribution functions (c.d.f.'s). We show that (for fixed r, p , and N) the hypergeometric c.d.f. starts out and remains below the corresponding binomial c.d.f. until it crosses at an integer close to the common mean and then remains above until both c.d.f.'s reach one at r . This observation sheds further light on the relationship between binomial and hypergeometric probabilities, C.I.'s, and c.d.f.'s.

In Schuster (1978), we showed that the lowest confidence level under the binomial case of Theorem 1.1

occurs when $r = 6$ with p approaching $5/6 - 1/\sqrt{6} \doteq 0.425085$ from the left. Here the confidence level percentage dips to nearly 91.011%. If the sample size r is fixed and the population size N tends to infinity in such a manner that the proportion of successes $p = n/N$ remains constant, then the hypergeometric value of $h(p)$ decreases to the binomial value of $h(p)$. Thus the 91.0% is also the best overall confidence level percentage one can achieve in both cases of Theorem 1.2. The lowest percentages again occur when $r = 6$ with N large and p approaching $5/6 - 1/\sqrt{6}$ from the left, i.e. when the hypergeometric model approaches the binomial model having the lowest confidence level.

In Section 3 we sketch a new proof of the numerical part of the proof of Theorem 1.1 which we use to produce Table 2. This table gives the lowest probability coverage of p by $\hat{p} \pm 1/\sqrt{r}$ for each $r \leq 200$ in either the binomial or hypergeometric case (we computed Table 2 entries to $r = 700$, but report only the first 200).

How conservative is the 91.0% confidence bound? The Central Limit Theorem tells us that for large r and (usually unknown) p not too close to 0 or 1, $\hat{p} \pm 1/\sqrt{r}$ is about a 95% C.I. for p . Table 2 suggests that as the sample size r becomes large, asymptotically the lowest confidence level coverage does indeed occur at $p = 1/2$ where the confidence coverage (0.954500) is given by the normal approximation to the binomial. However, except for the degenerate case $r = 1$, the lowest binomial probability coverage is below 0.9545 for every $r \leq 700$. There are only 2, 4, and 18 sample sizes where the probability coverage is below 0.92, 0.93, and 0.94, respectively. These 18 cases are listed in increasing order in Table 1. Note that the binomial probability coverage is at least 0.93 for $r \geq 13$ except for a value of 0.929369 at $r = 20$ and at least 0.940 for $r \geq 43$ except for a value of 0.939641 at $r = 56$. When considering theoretical models to approximate real world problems, there is often little practical difference between confidence levels in the range from 91% to 95%. Thus the significance of Theorems 1.1 and 1.2 are that they give a good, simple, easily stated accuracy rule, holding for all r , N , and p , which can easily be understood by the layman when assessing accuracy of estimates and discussing the relationship between accuracy and sample size.

2. Lowest Probability Coverages

Since 1978, available numerical methods in the form of exact arithmetic capabilities have improved substantially and we can now use an incomplete Beta representation of the tail of the binomial and the power of the software system *Mathematica* to give an alternate proof of the numerical part of the proof of Theorem 1.1. This proof

is of independent interest in that for any fixed sample size r it gives an algorithm for calculating the exact value of the infimum of

$$h(p) = h(p; r) = \text{Prob}(|\hat{p} - p| \leq 1/\sqrt{r})$$

as well as an exact location (and direction) where the infimum occurs. These results are given in Table 2 for $1 \leq r \leq 200$. This table has four entries: r , $\inf h(p; r)$, location p , d . Here, r is the sample size, $\inf h(p; r)$ is the infimum of $h(p; r)$ over $0 \leq p \leq 1$ and location p gives a point p where the infimum is realized. The d (for direction) heading indicates that the infimum occurs from the right(+) or left(-) at the given location p . A sketch of the argument justifying these table entries is contained in Section 3. We give the exact value of the location as well as a six digit approximation and round the infimum to six decimal digits. Note that the sequence $H_r = \inf\{h(p; r) : 0 \leq p \leq 1\}$ is not monotone in r as one might expect. For example, the lowest confidence level coverage of $\hat{p} \pm 1/\sqrt{r}$ for $r = 16$ is 0.950958 while it dips to 0.929369 at $r = 20$. This table suggests that as the sample size r becomes large, asymptotically the lowest confidence level occurs at $p = 1/2$ where the lowest confidence level (0.954500) is given by the normal approximation to the binomial. We have shown that the lowest probability coverage of p by $\hat{p} \pm 1/\sqrt{r}$ rounds to at least 0.94 in all three cases for $r \geq 21$. The coverage probability is above 0.940 for $r \geq 43$ except for the value 0.939641 at $r = 56$.

Table 1.
Probability Coverages Below 0.94

Sample size r	$\inf h(p; r)$
6	0.910113
2	0.914214
12	0.922074
3	0.924501
8	0.927673
20	0.929369
5	0.930495
15	0.932670
11	0.932957
30	0.934054
24	0.935862
19	0.935884
42	0.937285
4	0.937500
29	0.938338
35	0.938548
56	0.939641
7	0.939671

Table 2. Lowest Probability Coverage 1-50

r	$\inf h(p; r)$	location p	$\approx p$	d
1	1.0	0	0	
2	0.914214	$(2 - \sqrt{2})/2$	0.292893	-
3	0.924501	$(3 - \sqrt{3})/3$	0.42265	-
4	0.9375	1/2	0.5	+
5	0.930495	$(1)/\sqrt{5}$	0.447214	+
6	0.910113	$(5 - \sqrt{6})/6$	0.425085	-
7	0.939671	$(6 - \sqrt{7})/7$	0.479178	-
8	0.927673	$(1 + 2\sqrt{2})/8$	0.478553	+
9	0.948609	4/9	0.444444	-
10	0.941889	$(8 - \sqrt{10})/10$	0.483772	-
11	0.932957	$(2 + \sqrt{11})/11$	0.48333	+
12	0.922074	$(9 - 2\sqrt{3})/12$	0.461325	-
13	0.946211	$(10 - \sqrt{13})/13$	0.491881	-
14	0.940294	$(3 + \sqrt{14})/14$	0.481547	+
15	0.93267	$(11 - \sqrt{15})/15$	0.475134	-
16	0.950958	1/2	0.5	+
17	0.947235	$(4 + \sqrt{17})/17$	0.47783	+
18	0.941875	$(13 - 3\sqrt{2})/18$	0.48652	-
19	0.935884	$(5 + \sqrt{19})/19$	0.492574	+
20	0.929369	$(14 - 2\sqrt{5})/20$	0.476393	-
21	0.949209	$(15 - \sqrt{21})/21$	0.496068	-
22	0.94549	$(6 + \sqrt{22})/22$	0.485928	+
23	0.940946	$(16 - \sqrt{23})/23$	0.487138	-
24	0.935862	$(7 + \sqrt{24})/24$	0.495791	+
25	0.952342	12/25	0.48	-
26	0.949366	$(18 - \sqrt{26})/26$	0.496192	-
27	0.946167	$(8 + 3\sqrt{3})/27$	0.488746	+
28	0.942429	$(19 - 2\sqrt{7})/28$	0.489589	-
29	0.938338	$(9 + \sqrt{29})/29$	0.49604	+
30	0.934054	$(20 - \sqrt{30})/30$	0.484092	-
31	0.950826	$(21 - \sqrt{31})/31$	0.497814	-
32	0.948224	$(10 + 4\sqrt{2})/32$	0.489277	+
33	0.945192	$(22 - \sqrt{33})/33$	0.492589	-
34	0.941971	$(11 + \sqrt{34})/34$	0.495028	+
35	0.938548	$(23 - \sqrt{35})/35$	0.488112	-
36	0.952969	1/2	0.5	+
37	0.950883	$(24 - \sqrt{37})/37$	0.48425	-
38	0.948467	$(25 - \sqrt{38})/38$	0.495673	-
39	0.945959	$(13 + \sqrt{39})/39$	0.493461	+
40	0.94323	$(26 - \sqrt{40})/40$	0.491886	-
41	0.9403	$(14 + \sqrt{41})/41$	0.497637	+
42	0.937285	$(27 - \sqrt{42})/42$	0.488554	-
43	0.951798	$(28 - \sqrt{43})/43$	0.498664	-
44	0.949863	$(15 + 2\sqrt{11})/44$	0.491665	+
45	0.947686	$(29 - 3\sqrt{5})/45$	0.495373	-
46	0.945432	$(16 + \sqrt{46})/46$	0.495268	+
47	0.943038	$(30 - \sqrt{47})/47$	0.492433	-
48	0.940482	$(17 + 4\sqrt{3})/48$	0.498504	+
49	0.953398	24/49	0.489796	-
50	0.95172	$(32 - 5\sqrt{2})/50$	0.498579	-

Table 2. Lowest Probability Coverage 51-100

r	$\inf h(p; r)$	location p	$\approx p$	d
51	0.949993	$(18 + \sqrt{51})/51$	0.492969	+
52	0.94809	$(33 - 2\sqrt{13})/52$	0.49594	-
53	0.946128	$(19 + \sqrt{53})/53$	0.495851	+
54	0.94406	$(34 - \sqrt{54})/54$	0.493547	-
55	0.941872	$(20 + \sqrt{55})/55$	0.498476	+
56	0.939641	$(35 - \sqrt{56})/56$	0.491369	-
57	0.952429	$(36 - \sqrt{57})/57$	0.499126	-
58	0.950929	$(21 + \sqrt{58})/58$	0.493375	+
59	0.949284	$(37 - \sqrt{59})/59$	0.49693	-
60	0.947604	$(22 + 2\sqrt{15})/60$	0.495766	+
61	0.94583	$(38 - \sqrt{61})/61$	0.494914	-
62	0.943978	$(23 + \sqrt{62})/62$	0.497968	+
63	0.942077	$(39 - 3\sqrt{7})/63$	0.493059	-
64	0.953647	1/2	0.5	+
65	0.952348	$(40 - \sqrt{65})/65$	0.49135	-
66	0.950951	$(41 - \sqrt{66})/66$	0.498121	-
67	0.949527	$(25 + \sqrt{67})/67$	0.495304	+
68	0.948013	$(42 - 2\sqrt{17})/68$	0.496379	-
69	0.946453	$(26 + \sqrt{69})/69$	0.497197	+
70	0.944838	$(43 - \sqrt{70})/70$	0.494763	-
71	0.943147	$(27 + \sqrt{71})/71$	0.49896	+
72	0.941433	$(44 - 12\sqrt{2})/72$	0.49326	-
73	0.952861	$(45 - \sqrt{73})/73$	0.499397	-
74	0.951661	$(45 - \sqrt{74})/74$	0.49186	-
75	0.950373	$(46 - 5\sqrt{3})/75$	0.497863	-
76	0.949067	$(29 + 2\sqrt{19})/76$	0.496287	+
77	0.947695	$(47 - \sqrt{77})/77$	0.496429	-
78	0.946283	$(30 + \sqrt{78})/78$	0.497843	+
79	0.944831	$(48 - \sqrt{79})/79$	0.495086	-
80	0.943316	$(31 + 4\sqrt{5})/80$	0.499303	+
81	0.953833	40/81	0.493827	-
82	0.952757	$(50 - \sqrt{82})/82$	0.499325	-
83	0.951665	$(50 - \sqrt{83})/83$	0.492645	-
84	0.950503	$(51 - 2\sqrt{21})/84$	0.498034	-
85	0.949325	$(33 + \sqrt{85})/85$	0.496701	+
86	0.948096	$(52 - \sqrt{86})/86$	0.496818	-
87	0.946835	$(34 + \sqrt{87})/87$	0.498016	+
88	0.945541	$(53 - \sqrt{88})/88$	0.495672	-
89	0.944198	$(35 + \sqrt{89})/89$	0.499258	+
90	0.94284	$(54 - 3\sqrt{10})/90$	0.494591	-
91	0.953171	$(55 - \sqrt{91})/91$	0.499567	-
92	0.952188	$(55 - 2\sqrt{23})/92$	0.493569	-
93	0.951151	$(56 - \sqrt{93})/93$	0.498455	-
94	0.950102	$(37 + \sqrt{94})/94$	0.496759	+
95	0.949008	$(57 - \sqrt{95})/95$	0.497402	-
96	0.947891	$(38 + \sqrt{96})/96$	0.497895	+
97	0.946743	$(58 - \sqrt{97})/97$	0.496404	-
98	0.94556	$(39 + 7\sqrt{2})/98$	0.498974	+
99	0.94436	$(59 - 3\sqrt{11})/99$	0.495456	-
100	0.953956	1/2	0.5	+

Table 2. Lowest Probability Coverage 101-150

r	$\inf h(p; r)$	location p	$\approx p$	d
101	0.953077	$(60 - \sqrt{101})/101$	0.494556	-
102	0.952159	$(61 - \sqrt{102})/102$	0.499024	-
103	0.951232	$(41 + \sqrt{103})/103$	0.496591	+
104	0.950261	$(62 - \sqrt{104})/104$	0.498096	-
105	0.949278	$(42 + \sqrt{105})/105$	0.49759	+
106	0.948263	$(63 - \sqrt{106})/106$	0.497211	-
107	0.947225	$(43 + \sqrt{107})/107$	0.498543	+
108	0.946167	$(64 - 6\sqrt{3})/108$	0.496368	-
109	0.945075	$(44 + \sqrt{109})/109$	0.499452	+
110	0.943974	$(65 - \sqrt{110})/110$	0.495563	-
111	0.953401	$(66 - \sqrt{111})/111$	0.499679	-
112	0.952581	$(66 - 4\sqrt{7})/112$	0.494795	-
113	0.951726	$(67 - \sqrt{113})/113$	0.498848	-
114	0.950864	$(46 + \sqrt{114})/114$	0.497167	+
115	0.949969	$(68 - \sqrt{115})/115$	0.498054	-
116	0.949061	$(47 + 2\sqrt{29})/116$	0.49802	+
117	0.948129	$(69 - 3\sqrt{13})/117$	0.497294	-
118	0.947176	$(48 + \sqrt{118})/118$	0.498837	+
119	0.946209	$(70 - \sqrt{119})/119$	0.496565	-
120	0.945212	$(49 + \sqrt{120})/120$	0.49962	+
121	0.954053	60/121	0.495868	-
122	0.953306	$(72 - \sqrt{122})/122$	0.499628	-
123	0.952549	$(72 - \sqrt{123})/123$	0.495199	-
124	0.951763	$(73 - 2\sqrt{31})/124$	0.498907	-
125	0.95097	$(51 + 5\sqrt{5})/125$	0.497443	+
126	0.950151	$(74 - 3\sqrt{14})/126$	0.498215	-
127	0.94932	$(52 + \sqrt{127})/127$	0.498184	+
128	0.94847	$(75 - 8\sqrt{2})/128$	0.497549	-
129	0.947603	$(53 + \sqrt{129})/129$	0.498898	+
130	0.946722	$(76 - \sqrt{130})/130$	0.49691	-
131	0.945818	$(54 + \sqrt{131})/131$	0.499584	+
132	0.944908	$(77 - 2\sqrt{33})/132$	0.496295	-
133	0.953575	$(78 - \sqrt{133})/133$	0.499755	-
134	0.952882	$(78 - \sqrt{134})/134$	0.495703	-
135	0.952164	$(79 - \sqrt{135})/135$	0.499119	-
136	0.951442	$(56 + \sqrt{136})/136$	0.497514	+
137	0.950695	$(80 - \sqrt{137})/137$	0.498506	-
138	0.949941	$(57 + \sqrt{138})/138$	0.498169	+
139	0.949168	$(81 - \sqrt{139})/139$	0.497915	-
140	0.948383	$(58 + 2\sqrt{35})/140$	0.498301	+
141	0.947585	$(82 - \sqrt{141})/141$	0.497345	-
142	0.946768	$(59 + \sqrt{142})/142$	0.499411	+
143	0.945945	$(83 - \sqrt{143})/143$	0.496795	-
144	0.954123	1/2	0.5	+
145	0.95349	$(84 - \sqrt{145})/145$	0.496265	-
146	0.952839	$(85 - \sqrt{146})/146$	0.499431	-
147	0.952184	$(61 + 7\sqrt{3})/147$	0.497445	+
148	0.951506	$(86 - 2\sqrt{37})/148$	0.498882	-
149	0.950824	$(62 + \sqrt{149})/149$	0.498031	+
150	0.950124	$(87 - 5\sqrt{6})/150$	0.49835	-

Table 2. Lowest Probability Coverage 151-200

r	$\inf h(p; r)$	location p	$\approx p$	d
151	0.949416	$(63 + \sqrt{151})/151$	0.498597	+
152	0.948694	$(88 - \sqrt{152})/152$	0.497837	-
153	0.947959	$(64 + 3\sqrt{17})/153$	0.499146	+
154	0.947215	$(89 - \sqrt{154})/154$	0.49734	-
155	0.946455	$(65 + \sqrt{155})/155$	0.499077	+
156	0.94569	$(90 - 2\sqrt{39})/156$	0.496859	-
157	0.953711	$(91 - \sqrt{157})/157$	0.499809	-
158	0.953117	$(91 - \sqrt{158})/158$	0.496394	-
159	0.952506	$(92 - \sqrt{159})/159$	0.499311	-
160	0.951891	$(67 + \sqrt{160})/160$	0.497807	+
161	0.951258	$(93 - \sqrt{161})/161$	0.498829	-
162	0.950621	$(68 + 9\sqrt{2})/162$	0.498321	+
163	0.949969	$(94 - \sqrt{163})/163$	0.498361	-
164	0.949309	$(69 + 2\sqrt{41})/164$	0.498819	+
165	0.948639	$(95 - \sqrt{165})/165$	0.497908	-
166	0.947956	$(70 + \sqrt{166})/166$	0.497302	+
167	0.947268	$(96 - \sqrt{167})/167$	0.497468	-
168	0.946563	$(71 + \sqrt{168})/168$	0.499771	+
169	0.95418	84/169	0.497041	-
170	0.95363	$(98 - \sqrt{170})/170$	0.499774	-
171	0.953076	$(98 - 3\sqrt{19})/171$	0.496628	-
172	0.952507	$(99 - 2\sqrt{43})/172$	0.499332	-
173	0.951934	$(73 + \sqrt{173})/173$	0.497994	+
174	0.951347	$(100 - \sqrt{174})/174$	0.498903	-
175	0.950755	$(74 + 5\sqrt{7})/175$	0.49845	+
176	0.950151	$(101 - 4\sqrt{11})/176$	0.498486	-
177	0.94954	$(75 + \sqrt{177})/177$	0.498893	+
178	0.94892	$(102 - \sqrt{178})/178$	0.498081	-
179	0.94829	$(76 + \sqrt{179})/179$	0.499325	+
180	0.947654	$(103 - 6\sqrt{5})/180$	0.497687	-
181	0.947005	$(77 + \sqrt{181})/181$	0.499744	+
182	0.946354	$(104 - \sqrt{182})/182$	0.497304	-
183	0.95382	$(105 - \sqrt{183})/183$	0.499848	-
184	0.953305	$(105 - \sqrt{184})/184$	0.496931	-
185	0.952777	$(106 - \sqrt{185})/185$	0.499452	-
186	0.952247	$(79 + \sqrt{186})/186$	0.498055	+
187	0.951704	$(107 - \sqrt{187})/187$	0.499065	-
188	0.951157	$(80 + 2\sqrt{47})/188$	0.498464	+
189	0.9506	$(108 - \sqrt{189})/189$	0.498689	-
190	0.950036	$(81 + \sqrt{190})/190$	0.498863	+
191	0.949465	$(109 - \sqrt{191})/191$	0.498323	-
192	0.948885	$(82 + 8\sqrt{3})/192$	0.499252	+
193	0.9483	$(110 - \sqrt{193})/193$	0.497967	-
194	0.947704	$(83 + \sqrt{194})/194$	0.499631	+
195	0.947105	$(111 - \sqrt{195})/195$	0.497619	-
196	0.954223	1/2	0.5	+
197	0.953747	$(112 - \sqrt{197})/197$	0.497281	-
198	0.95326	$(113 - 3\sqrt{22})/198$	0.49964	-
199	0.952772	$(113 - \sqrt{199})/199$	0.496951	-
200	0.952271	$(114 - 10\sqrt{2})/200$	0.499289	-

3. Proof for Table Entries

Let the random variable X be binomially distributed with parameters r and p . Then the incomplete Beta representation of the right tail probabilities of X is: $f(k; r, p) = P(X \geq k) = \beta(k, r - k + 1)^{-1} \int_0^p t^{k-1} (1-t)^{r-k} dt$. But then

$$h(p) = h(p; r) = \text{Prob}(|\hat{p} - p| \leq 1/\sqrt{r})$$

can be written in the form $h(p) = h(p; r, x, y) = P(x \leq X \leq y) = P(X \geq x) - P(X \geq y + 1)$ for $x = x(p) = \text{Ceiling}(rp - \sqrt{r})$ and $y = y(p) = \text{Floor}(rp + \sqrt{r})$. For fixed r , the function $h(p)$ uses the same x and y for all p in some interval, say $I(p)$, and is therefore continuous and differentiable as a function of p except possibly at the endpoints of the intervals $I(p)$ where the x and/or y might change. These change points p belong to the set $E = E(r) = \{p | p = (k \pm \sqrt{r})/r, k = 0, 1, \dots, r\} \cap [0, 1]$. Using this Beta representation, one can take the derivative of $h(p)$ on any interval $I(p)$ and show that $h(p)$ is concave down on each of the intervals $I(p)$. Thus the infimum of $h(\cdot)$ will occur from the left or right at one of the points in E .

The infimum of $h(\cdot)$ over the finite set E can be computed as follows. Let $b(x) = b(x; r, p) = P(X = x; r, p)$ be the probability mass function of the binomial with parameters r and p . Then the left and right hand limits of $h(\cdot)$ at p are of the form $h(p \pm) = h(p) - j(p)$ where the jump function $j(p)$ is given in Table 3 for the various cases. The entries in the table are obtained by taking the left and right limits of $\text{Ceiling}(l)$ and $\text{Floor}(u)$ where $l = l(p) = rp - \sqrt{r}$ and $u = u(p) = rp + \sqrt{r}$. There are four cases depending on whether or not l and u are integers. For example, consider the case when $l = l(p)$ is integer, but $u(p)$ is not. Then both the left and right hand limits of $y(\cdot)$ at p equal $\text{Floor}(rp + \sqrt{r}) = \text{Floor}(u(p)) = y(p)$. Moreover, $x(p+) = \text{Ceiling}((rp - \sqrt{r})+) = \text{Ceiling}(l(p+)) = \text{Ceiling}(l(p)) + 1 = x(p) + 1 = l + 1$, but $x(p-) = x(p) = l$. Thus $h(p+) = h(p) - b(l)$ and $h(p-) = h(p)$ and the corresponding jump functions are $b(l)$ and 0 as given in the last row of Table 3. The remaining 6 cases in Table 3 follow in similar fashion. The important point is that we have exact expressions for both $h(p+)$ and $h(p-)$. The Table 3 column headed by d (for direction) uses $+$ and $-$ to designate the appropriate $j(p)$ function corresponding to limits from the right($+$) and left($-$), respectively.

The software system *Mathematica* was used to program the entries in Table 3 to compute the exact and approximate value of $\inf\{h(p; r) | 0 \leq p \leq 1\}$ as well as an the exact and approximate point p where the infimum was realized for all $r \leq 700$. A summary for $r \leq 200$ is given in Table 2. A numerical check of this routine was

made by computing the minimum of $h(p; r)$ for $r \leq 100$ over a grid with spacing $s = 0.0001$.

As general guidance, picking r to be a perfect square seems to do quite well. For r a perfect square larger than one, the probability coverage increases with r (at least to 700), has the lowest value of 0.9375 at $r = 4$, a value of 0.948609 at $r = 9$, and is above 0.95 for squares $r \geq 16$ (0.95 is within 0.0045 of the upper asymptotic value!). See the Table 4 summary for squares below 700. It also appears (check the entries in Table 2) that if r is a perfect square, then $\inf h(p; r)$, the lowest probability coverage at r , is larger than $\inf h(p; r_0)$ for any integer r_0 , $1 < r_0 < r$.

Table 3. Form of Jump Function $j(p)$

	d	l integer	l not integer
u integer	$-$	$b(u)$	$b(u)$
	$+$	$b(l)$	0
u not integer	$-$	0	0
	$+$	$b(l)$	0

Table 4.
Coverages for Perfect Squares

r	$\inf h(p; r)$	r	$\inf h(p; r)$
1	1.0	196	0.954223
4	0.937500	225	0.954260
9	0.948609	289	0.954313
16	0.950958	324	0.954333
25	0.952342	361	0.954350
36	0.952969	400	0.954365
49	0.953398	441	0.954377
64	0.953647	484	0.954388
81	0.953833	529	0.954398
100	0.953956	576	0.954406
121	0.954053	625	0.954413
144	0.954123	676	0.954420
169	0.954180	∞	0.954500

4. References

- Johnson, N.L. and Kotz, S. (1969), *Discrete Distributions*. Houghton Mifflin, Boston.
- Schuster, E.F. (1978), "On the accuracy of probability estimates". *Mathematics Magazine*, 51, 227-229.
- Schuster, E.F. (1992), "Accuracy of proportion estimators", preprint.
- Uhlmann, V.W. (1966), "Vergleich der hypergeometrischen mit der binomial-verteilung". *Metrika*, 10, 145-158.

Simulating Gaussian Random Processes with Specified Spectra

Donald B. Percival
Applied Physics Laboratory, HN-10
University of Washington
Seattle, WA 98195

Abstract

Abstract—We discuss the problem of generating realizations of length N from a Gaussian stationary process $\{Y_t\}$ with a specified spectral density function $S_Y(\cdot)$. We review three methods for generating the required realizations and consider their relative merits. In particular, we discuss an approximate frequency domain technique that is evidently used frequently in practice, but that has some potential pitfalls. We discuss extensions to this technique that allow it to be used to generate realizations from a power-law process with spectral density function similar to $S(f) = |f|^\alpha$ for $\alpha < 0$.

I. Introduction

Let $\{Y_t\}$ be a real-valued Gaussian stationary process with spectral density function (sdf) $S_Y(\cdot)$, autocorrelation sequence (acvs) $\{s_{\tau,Y}\}$ and zero mean. If we define the sampling time between observations Y_t and Y_{t+1} to be unity so that the Nyquist frequency is $\frac{1}{2}$, then the acvs is related to the sdf via the usual relationship

$$s_{\tau,Y} = \int_{-\frac{1}{2}}^{\frac{1}{2}} S_Y(f) e^{i2\pi f\tau} df, \quad \text{where } i \equiv \sqrt{-1}. \quad (1)$$

A problem of considerable practical interest is to generate a sample of length N of this process (i.e., a realization of Y_0, \dots, Y_{N-1}) on a digital computer by suitably transforming samples of a zero mean, unit variance Gaussian white noise process $\{W_t\}$. In this paper we discuss three methods for doing this: an exact time domain method that is valid for all sdf's (Section II), an exact frequency domain method that is valid only for some sdf's (Section III), and an approximate frequency domain method that can be used for all sdf's (Section IV). We place particular emphasis on generating time series from stationary and nonstationary power-law (long memory) processes (Sections V and VI). (The three methods we discuss here are certainly not the only ones that have been advocated in the literature—one important omission is an approximate time domain method based upon the class of autoregressive-moving average models.)

II. An Exact Time Domain Method

If the acvs $\{s_{\tau,Y}\}$ is readily known out to lag $N-1$, there are well-known time domain techniques for generating samples of $\{Y_t\}$ (see, for example, Franklin, 1965). Typically these involve an lower-upper Cholesky factorization of the inverse of the N -th order Toeplitz covariance matrix for Y_0, \dots, Y_{N-1} (see Demeure and Scharf, 1987, for a good review). This factorization can be accomplished using the Levinson-Durbin recursions and then used to generate the desired samples, as follows. Let W_0, \dots, W_{N-1} be a set of N independent and identically distributed Gaussian random variables (rv's) with zero mean and unit variance. With $Y_0 = \sigma_0 W_0$, we generate the $N-1$ remaining samples recursively via

$$Y_t = \sum_{j=1}^t \phi_{j,t} Y_{t-j} + W_t \sigma_t, \quad t = 1, \dots, N-1.$$

The σ_t 's and $\phi_{j,t}$ are obtained by first setting $\sigma_0^2 = s_{0,Y}$ and then recursively computing for $t = 1, \dots, N-1$

$$\begin{aligned} \phi_{t,t} &= \frac{s_{t,Y} - \sum_{j=1}^{t-1} \phi_{j,t-1} s_{t-j,Y}}{\sigma_{t-1}^2} \\ \phi_{j,t} &= \phi_{j,t-1} - \phi_{t,t} \phi_{t-j,t-1}, \quad 1 \leq j \leq t-1 \\ \sigma_t^2 &= \sigma_{t-1}^2 (1 - \phi_{t,t}^2) \end{aligned}$$

(for $t = 1$ the summation in the first equation is taken to be 0, and the second equation is skipped).

There are two potential drawbacks to this exact method. First, once we have computed the σ_t 's and $\phi_{j,t}$'s, the number of floating point operations needed to generate a sample of length N is $O(N^2)$. There are ways, however, of reducing this number to $O(N)$ if in fact the Toeplitz matrix possesses enough special structure (this is the case if, for example, $\{Y_t\}$ is an autoregressive-moving average process—see Kay, 1981, for details). Second, if we are given the sdf $S_Y(\cdot)$ instead of the acvs, we must first obtain the required $s_{\tau,Y}$'s. In principle this can always be done via numerical integration, but in practice this approach can be error-prone and time consuming.

III. An Exact Frequency Domain Method

Davies and Harte (1987) recently outlined a frequency domain technique for simulating $\{Y_t\}$ that makes use of a fast Fourier transform (fft) algorithm and hence requires only $O(N \log(N))$ operations. As these authors noted, their method is not completely general in that it can fail to work for some processes. The situations for which their method is applicable are easily described by a nonnegativity constraint. Their method has in fact appeared previously in the literature in the context of simulating Gaussian moving average processes of order q , for which the nonnegativity constraint holds as long as N is greater than q (see Davis, Hagan, and Borgman, 1981, and the discussion of their work in Ripley, 1987).

Let M be any even positive integer (typically a power of 2), and define $f_j = \frac{j}{M}$. Let

$$S_j \equiv \sum_{\tau=-(\frac{M}{2}-1)}^{\frac{M}{2}-1} s_{\tau,Y} e^{-2\pi f_j \tau}, \quad 0 \leq j \leq \frac{M}{2}. \quad (2)$$

Note that we can rewrite the above as

$$S_j = \left(\sum_{\tau=0}^{\frac{M}{2}-1} s_{\tau,Y} e^{-2\pi f_j \tau} + \sum_{\tau=\frac{M}{2}+1}^{M-1} s_{M-\tau,Y} e^{-2\pi f_j \tau} \right),$$

so we can obtain the S_j 's via the discrete Fourier transform of the following sequence of length M :

$$s_{0,Y}, s_{1,Y}, \dots, s_{\frac{M}{2}-1,Y}, 0, s_{\frac{M}{2}-1,Y}, s_{\frac{M}{2}-2,Y}, \dots, s_{1,Y}.$$

We can also reexpress Equation (2) as

$$S_j = \int_{-\frac{1}{2}}^{\frac{1}{2}} W(f_j - f) S_Y(f) df,$$

where

$$W(f) \equiv \frac{\sin((M-1)\pi f)}{\sin(\pi f)}.$$

Because $W(\cdot)$ oscillates between positive and negative values, it is possible that some of the S_j 's are negative. Note, however, that, if $\{Y_t\}$ were a moving average process of order $q \leq \frac{M}{2} - 1$ so that $s_{\tau,Y} = 0$ for $|\tau| \geq \frac{M}{2}$, then we would have $S_j = S_Y(f_j)$ so that $S_j \geq 0$. In order for the simulation method to work, we must impose the nonnegativity constraint that $S_j \geq 0$ for $0 \leq j \leq \frac{M}{2}$.

Let W_0, \dots, W_{M-1} be a set of M independent and identically distributed Gaussian rv's with zero mean and unit variance. Define

$$V_j \equiv \begin{cases} \sqrt{S_0} W_0, & j = 0; \\ \sqrt{\frac{1}{2} S_j} (W_{2j-1} + i W_{2j}), & 1 \leq j < \frac{M}{2}; \\ \sqrt{S_{\frac{M}{2}}} W_{M-1}, & j = \frac{M}{2}; \\ V_{M-j}^*, & \frac{M}{2} < j \leq M-1 \end{cases}$$

(the asterisk denotes complex conjugation). Note that

$$\text{cov}\{V_j, V_k\} = E\{V_j^* V_k\} = \begin{cases} S_j, & \text{if } j = k; \\ 0, & \text{otherwise.} \end{cases}$$

We next define the process $\{V_t\}$ via

$$V_t \equiv \frac{1}{\sqrt{M}} \sum_{j=0}^{M-1} V_j e^{-i2\pi f_j t}, \quad t = 0, \dots, M-1. \quad (3)$$

By construction, the process $\{V_t\}$ is real-valued. Because V_t is a linear combination of Gaussian rv's, the process is Gaussian. A straight-forward exercise shows that $\{V_t\}$ is a stationary process with zero mean and acvs $\{s_{\tau,V}\}$ given by

$$s_{\tau,V} = \frac{1}{M} \sum_{j=0}^{M-1} S_j e^{i2\pi f_j \tau}. \quad (4)$$

In contrast to $\{Y_t\}$, however, the stationary process $\{V_t\}$ is a harmonic process; i.e., it does not possess an sdf, but its spectral properties are given by an integrated spectrum that is a step function with steps at the $\pm f_j$'s. This fact implies that realizations of $\{V_t\}$ are periodic with period M , and hence so is its acvs $\{s_{\tau,V}\}$. Note that, if M is a power of 2, we can readily compute both $\{V_t\}$ and $\{s_{\tau,V}\}$ using a conventional fft algorithm.

By substituting the definition for S_j in Equation (2) into Equation (4) and interchanging the order of the two summations, we obtain

$$s_{\tau,V} = \frac{1}{M} \sum_{\rho=-(\frac{M}{2}-1)}^{\frac{M}{2}-1} s_{\rho,Y} \left(\sum_{j=0}^{M-1} e^{-i2\pi f_j(\tau-\rho)} \right).$$

From the result

$$\sum_{j=0}^{M-1} e^{-i2\pi f_j \eta} = \begin{cases} M, & \text{for } \eta = 0, \pm M, \pm 2M, \dots; \\ 0, & \text{otherwise,} \end{cases}$$

we obtain

$$s_{\tau,V} = s_{\tau,Y} \text{ for all } |\tau| \leq \frac{M}{2}.$$

Hence the statistical properties of V_0, \dots, V_{N-1} are identical to those of Y_0, \dots, Y_{N-1} if we set $N \leq \frac{M}{2}$.

As is true for the exact time domain method, we might need to obtain the required $s_{\tau,Y}$'s via numerical integration if we are given the sdf $S_Y(\cdot)$ instead of the acvs. Also, because of the nonnegativity constraints on the S_j 's, this method cannot be used for all stationary processes. As we noted previously, it does work for moving average processes of order $q \leq \frac{M}{2} - 1$. Since every nonnegative lag window spectral estimate has an sdf corresponding to that of a high order moving average process, this exact frequency domain method is useful for simulating time series from such spectral estimates.

IV. An Approximate Frequency Domain Method

We consider here the construction of a zero mean Gaussian process $\{U_t\}$ whose acvs $\{s_{\tau,U}\}$ agrees—to a good approximation—with $\{s_{\tau,Y}\}$ out to lag $N-1$. To begin with, we make the assumption that the sdf $S_Y(\cdot)$ is continuous over $[-\frac{1}{2}, \frac{1}{2}]$ (we relax this restriction in the next section). Let M be any even integer greater than or equal to the desired sample size N . Let W_j , $j = 0, \dots, M-1$, be a set of M independent and identically distributed Gaussian rv's with zero mean and unit variance. Let $f_j \equiv \frac{j}{M}$ as before. We define the process $\{U_t\}$ via

$$U_t \equiv \frac{1}{\sqrt{M}} \sum_{j=0}^{M-1} U_j e^{-i2\pi f_j t}, \quad t = 0, \dots, M-1, \quad (5)$$

where

$$U_j \equiv \begin{cases} \sqrt{S_Y(0)}W_0, & j = 0; \\ \sqrt{\frac{1}{2}S_Y(f_j)}(W_{2j-1} + iW_{2j}), & 1 \leq j < \frac{M}{2}; \\ \sqrt{S_Y(\frac{1}{2})}W_{M-1}, & j = \frac{M}{2}; \\ U_{M-j}^*, & \frac{M}{2} < j \leq M-1. \end{cases}$$

By construction, $\{U_t\}$ is a real-valued Gaussian process. A straight-forward exercise shows that $\{U_t\}$ is a stationary process with zero mean and acvs $\{s_{\tau,U}\}$ given by

$$s_{\tau,U} = \frac{1}{M} \sum_{j=0}^{M-1} S(f_j) e^{i2\pi f_j \tau} \quad (6)$$

(note that this acvs can readily be computed using an fft). In contrast to $\{Y_t\}$, however, the stationary process $\{U_t\}$ is a harmonic process (as was the case with the V_t 's in Equation (3)).

Because the right-hand side of Equation (6) can be regarded as a Riemann sum approximation to the integral of Equation (1), we have $s_{\tau,U} \approx s_{\tau,Y}$ for possibly some values of τ , but certainly not all: whereas $\{s_{\tau,U}\}$ is periodic, $\{s_{\tau,Y}\}$ must damp down to 0. Note that, if we let M be a power of 2, Equation (5) can be quickly computed using a conventional fft algorithm—realizations of $\{U_t\}$ of length M can thus be readily generated. By making M large enough, we can make $s_{\tau,U}$ arbitrarily close to $s_{\tau,Y}$ for $\tau = 0, \dots, N-1$, and hence the simulations of U_0, \dots, U_{N-1} should have statistical properties that closely match those of Y_0, \dots, Y_{N-1} .

This scheme was evidently first proposed by Thompson (1973), who advocated just letting $M = N$. This formulation is in common use in the physical sciences,

perhaps due to the following result. Consider the periodogram of U_0, \dots, U_{M-1} :

$$\hat{S}_U^{(p)}(f) \equiv \frac{1}{M} \left| \sum_{t=0}^{M-1} U_t e^{-i2\pi f t} \right|^2. \quad (7)$$

If M is a power of 2, we can evaluate the periodogram quickly over the grid of Fourier frequencies f_j using an fft algorithm. When $M = N$, we have $E\{\hat{S}_U^{(p)}(f_j)\} = S_Y(f_j)$ by construction—hence realizations of $\{U_t\}$ have a periodogram in good apparent agreement with the target sdf $S_Y(\cdot)$ at the Fourier frequencies. Unfortunately, the periodogram for Y_0, \dots, Y_{M-1} can be a badly biased estimator of $S_Y(\cdot)$, so this intuitively pleasing agreement is misleading. Figure 1 illustrates this important point.

Mitchell and McPherson (1981) also discussed this approximate scheme. To avoid the " $M = N$ " problem discussed above, they advocated that M should be made larger than N commensurate with the "correlation length" of $\{Y_t\}$, but—beyond this brief statement—they did not provide explicit guidelines for selecting M relative to N . We can do so by noting the following useful measure of how well the $s_{\tau,U}$'s approximate the $s_{\tau,Y}$'s for $|\tau| \leq N-1$. Let $s_{\tau,U}^{(M)}$ denote the value of $s_{\tau,U}$ generated using Equation (6) for a particular value of M . Define $S_U^{(M)}(\cdot)$ as the function whose Fourier coefficients are equal to $s_{\tau,U}^{(M)}$ for $|\tau| \leq N-1$ and equal to $s_{\tau,Y}$ for $|\tau| \geq N$. Parseval's theorem then tells us that

$$\begin{aligned} SS(M) &\equiv \sum_{\tau=-(N-1)}^{N-1} |s_{\tau,U}^{(M)} - s_{\tau,Y}|^2 \\ &= \int_{-\frac{1}{2}}^{\frac{1}{2}} |S_U^{(M)}(f) - S_Y(f)|^2 df. \end{aligned}$$

$SS(M)$ can be interpreted as the average squared difference between the sdf $S_Y(\cdot)$ and the function $S_U^{(M)}(\cdot)$. As M gets large, $SS(M)$ will decrease to zero. If we know the $s_{\tau,Y}$'s, we can readily compute $SS(M)$ and find a value of M such that $S_U^{(M)}(\cdot)$ is sufficiently close to $S_Y(\cdot)$ in terms of average squared difference. If the $s_{\tau,Y}$'s cannot be readily computed, we can increase M by, say, factors of 2 until

$$\sum_{\tau=-(N-1)}^{N-1} |s_{\tau,U}^{(M)} - s_{\tau,U}^{(2M)}|^2$$

is small, indicating that the effect of doubling M is small in that the average squared difference between $S_U^{(M)}(\cdot)$ and $S_U^{(2M)}(\cdot)$ is small. Figure 2 illustrates how increasing M yields a better approximation to $\{s_{\tau,Y}\}$.

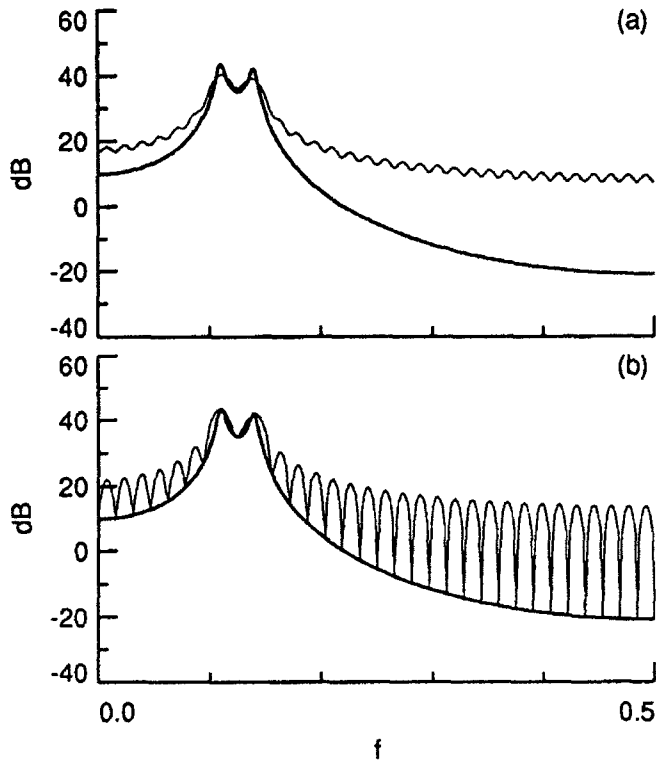


Figure 1. The thick curves in plots (a) and (b) show the true sdf $S_Y(\cdot)$ (on a decibel scale) for a particular stationary process $\{Y_t\}$. The thin bumpy curve in plot (a) shows the expected value of the periodogram for a sample of size $N = 64$ from this process. The thin bumpy curve in plot (b) shows the expected value of the periodogram $\hat{S}_U^{(p)}(\cdot)$ of Equation (7) for $M = N = 64$. If the statistical properties of $\{U_t\}$ closely matched those of $\{Y_t\}$, there would be good agreement between the two thin bumpy curves, but in fact they are substantially different. In particular, note that in plot (b) $E\{\hat{S}_U^{(p)}(f)\} = S_Y(f)$ for $f = f_j = \frac{j}{64}$ and $j = 0, \dots, 32$, whereas $E\{\hat{S}_Y^{(p)}(\cdot)\}$ and $S_Y(\cdot)$ in plot (a) differ at some of these frequencies by more than 2 orders of magnitude (20 dB).

V. Stationary Power-Law Processes

Suppose now that $\{Y_t\}$ is a stationary power-law process, which—by definition—has an sdf given by

$$S_Y(f) = |f|^\alpha S_0(f), \quad |f| \leq \frac{1}{2}, \quad (8)$$

where $-1 < \alpha < 0$, and $S_0(\cdot)$ is strictly positive, continuous and has bounded variation. A specific example of such a process is a fractional difference process with sdf

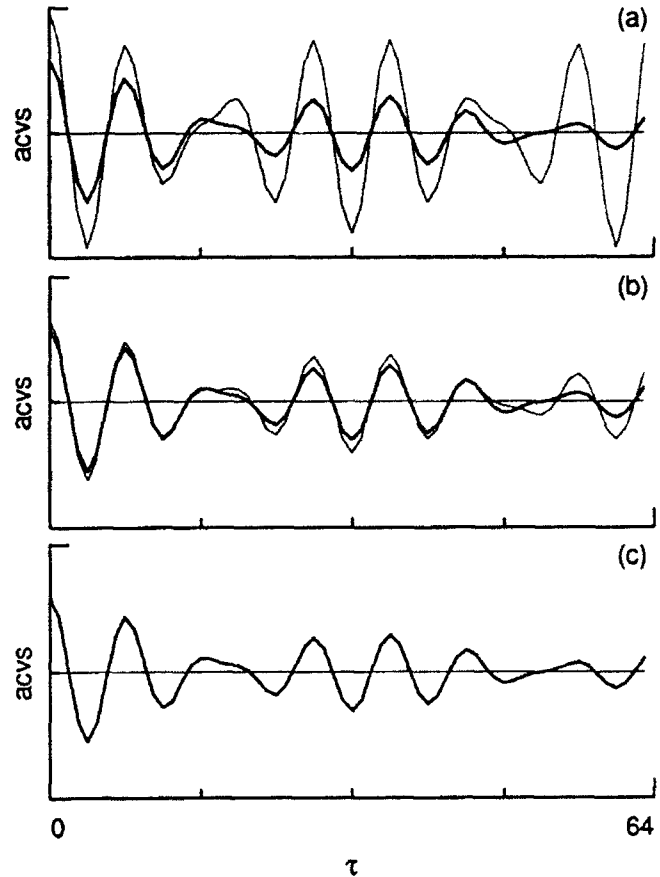


Figure 2. The thick curves in all three plot show the true acvs $\{s_{\tau,Y}\}$ for lags 0 to 63 for a particular stationary process $\{Y_t\}$ (in fact, the same process as was used in Figure 1). The thin curves show $\{s_{\tau,U}\}$ of Equation (6) for, from top to bottom, $M = 64, 128$ and 256 —the thin curve in plot (c) agrees so well with the thick curve that they are visibly indistinguishable.

given by

$$S_Y(f) = (2|\sin(\pi f)|)^\alpha, \quad |f| \leq \frac{1}{2}.$$

This process also has an acvs $\{s_{\tau,Y}\}$ that can be computed recursively by a simple formula. Using these easily computed $s_{\tau,Y}$'s, Davies and Harte (1987) found that the exact frequency domain method can be used to simulate fractional difference processes. Since a similar simple formula for the acvs is not readily available for other stationary power-law processes, it is of some interest to see if we can modify the approximate frequency domain method for use here. The main difficulty is that this method requires that $S_Y(0)$ be finite, whereas Equation (8) tells us that $S_Y(0) = \infty$. We thus need to find a suitable replacement for $S_Y(0)$.

To do so, let us return momentarily to the setup of the previous section, namely, a process with an sdf that is continuous over $[-\frac{1}{2}, \frac{1}{2}]$. An easy exercise tells us that

$$\bar{U} \equiv \frac{1}{M} \sum_{i=0}^{M-1} U_i = \frac{U_0}{\sqrt{M}} = \frac{\sqrt{S_Y(0)W_0}}{\sqrt{M}},$$

from which we obtain

$$\text{var}\{\bar{U}\} = \frac{S_Y(0)}{M} \approx \text{var}\{\bar{Y}\}, \text{ where } \bar{Y} = \frac{1}{M} \sum_{i=0}^{M-1} Y_i$$

(Priestley, 1981, p. 320). We can thus regard $S_Y(0)$ as an approximation to $M \cdot \text{var}\{\bar{Y}\}$. Künsch (1991) shows that, for a stationary power-law process $\{Y_i\}$,

$$\text{var}\{\bar{Y}\} \approx \frac{4S_0(0)\Gamma(1+\alpha)\sin(-\pi\alpha/2)}{(2\pi M)^{1+\alpha}(\alpha-1)} \equiv C_M.$$

This result suggests that we let $U_0 = \sqrt{MC_M}W_0$ for stationary power-law processes. Limited tests to date indicate that this substitution is effective. More work needs to be done, however, to find the best U_0 so that the statistical properties of U_0, \dots, U_{N-1} are as close as possible to those of Y_0, \dots, Y_{N-1} .

VI. Nonstationary Power-Law Processes

Suppose now that $\{Y_i\}$ is a nonstationary power-law process with an "sdf" given by Equation (8) with $\alpha \leq -1$ (because $S_Y(\cdot)$ integrates to ∞ , it is not a proper sdf). Processes such as these are common models in the physical sciences—some typical values for α are -1 ("flicker" noise), $-\frac{5}{3}$ (certain types of turbulence) and -2 ("random walk" noise). Yaglom (1958) developed a rigorous interpretation for the sdf $S_Y(\cdot)$ in terms of finite differences of $\{Y_i\}$. For example, if $-3 < \alpha \leq -1$, the first difference process $X_i \equiv Y_i - Y_{i-1}$ is a stationary process with sdf

$$S_X(f) = 4\sin^2(\pi f)S_Y(f), \quad |f| \leq \frac{1}{2},$$

an equation which is used to define $S_Y(\cdot)$ (for $\alpha \leq -3$, we define $S_Y(\cdot)$ using an appropriate higher order difference).

Because we define $\{Y_i\}$ for $-3 < \alpha \leq -1$ in terms of its first difference process $\{X_i\}$, we can simulate $\{X_i\}$ and form cumulative sums to simulate $\{Y_i\}$. Here we merely note that we can use $S_Y(\cdot)$ directly in the approximate frequency domain method because

$$U_i - U_{i-1} = \frac{-i}{\sqrt{M}} \sum_{j=0}^{M-1} 2\sin(\pi f_j)U_j e^{-i2\pi f_j(i-\frac{1}{2})},$$

approximates $Y_i - Y_{i-1}$ properly (again, care must be taken at $f = 0$).

VII. Concluding Comments

We have described three methods for simulating a stationary Gaussian process $\{Y_i\}$ with a specified sdf $S_Y(\cdot)$. If the $s_{r,Y}$'s for the process are readily available, then the best choice is the exact frequency domain method if the S_j 's of Equation (2) are in fact nonnegative. If the $s_{r,Y}$'s are not readily available or if one or more of the S_j 's are negative, then—with care—the approximate frequency domain method can be useful. However, the " $M = N$ " formulation of this method that is sometimes used should be avoided.

Acknowledgement

The author thanks the Office of Naval Research for support under contract number N00014-81-K-0095.

References

- Davies, R. B. and Harte, D. S. (1987) Tests for Hurst Effect. *Biometrika*, 74, 95-101.
- Davis, B. M., Hagan, R. and Borgman, L. E. (1981) A Program for the Finite Fourier Transform Simulation of Realizations from a One-Dimensional Random Function with Known Covariance. *Comp. and Geosci.*, 7, 199-206.
- Demeure, C. J. and Scharf, L. L. (1987) Linear Statistical Models for Stationary Sequences and Related Algorithms for Cholesky Factorization of Toeplitz Matrices. *IEEE Trans. Acoust., Speech, Signal Processing*, 35, 29-42.
- Franklin, J. N. (1965) Numerical Simulation of Stationary and Nonstationary Gaussian Random Processes. *SIAM Review*, 7, 68-80.
- Kay, S. M. (1981) Efficient Generation of Colored Noise. *Proc. IEEE*, 69, 480-481.
- Künsch, H. R. (1991) Dependence Among Observations: Consequences and Methods to Deal with It. In *Directions in Robust Statistics and Diagnostics, Part I*, edited by W. Stahel and S. Weisberg, New York: Springer-Verlag, 131-140.
- Mitchell, R. L. and McPherson, D. A. (1981) Generating Nonstationary Random Sequences. *IEEE Trans. Aero., Elec. Sys.*, 17, 553-560.
- Priestley, M. B. (1981) *Spectral Analysis and Time Series*. London: Academic Press.
- Ripley, B. D. (1987) *Stochastic Simulation*. New York: John Wiley & Sons.
- Thompson, R. (1973) Generation of Stochastic Processes with Given Spectrum. *Util. Math.*, 3, 127-137.
- Yaglom, A. M. (1958) Correlation Theory of Processes with Random Stationary n th Increments. *Translations Amer. Math. Soc.*, 8, 87-141.

Fast Algorithms for Modeling Rapidly Sampled Data*

H. Vincent Poor
Department of Electrical Engineering
Princeton University
Princeton, NJ 08544

Abstract

The problem of fitting linear recursive models to signals that are sampled at high rates relative to their underlying dynamics is considered. This time-series modeling regime is of particular interest in applications such as digital feedback control, wideband communications, and high-definition video, in which high relative sampling rates are often dictated by system stability or format considerations rather than by signal-processing needs. Most traditional signal processing algorithms are inherently ill-conditioned when applied in such situations. Recent work has shown that a representation of discrete-time signal dynamics based on incremental difference operators can lead to alternative algorithms that are both numerically stable and computationally efficient in this rapid-sampling regime. This methodology is reviewed here, with the *Levinson* or autoregressive modeling problem serving as a paradigm.

1. Introduction

Several emerging signal processing applications involve the modeling of discrete-time series that are derived through rapid sampling of continuous-time processes. Such applications include wideband communications, digital feedback control, and high-definition video transmission, in which high relative sampling rates are often dictated by format or system stability considerations. Many traditional procedures for fitting models, such as autoregressions, to time series are poorly suited for data obtained by sampling continuous-time processes at rates that are rapid relative to the dynamics of the underlying continuous processes. For example, when the *Levinson-Durbin* or *Schur* algorithm is used to estimate the parameters of the standard autoregressive (AR) model, large computational errors can occur due to the ill-conditioning of the Toeplitz covariance matrix of data in this high-speed regime. Considerable recent progress toward ameliorating such problems has been made through the use of an incremen-

tal divided-difference operator, rather than the conventional shift operator, to represent the dynamics of sampled data. This approach leads to an alternative signals calculus that allows for a unification of continuous and discrete time formulations, enables a smooth transition from sampled-data algorithms to their continuous-time counterparts, and consequently enhances the numerical conditioning of algorithms in the high-speed regime. For example, in the AR-modeling problem, as the sampling period goes to zero, the coefficients from the divided-difference model converge to certain regression parameters that depend directly on the statistics of the continuous-time process. This phenomenon is in contrast with the standard AR parameters, which converge to meaningless limits in this regime.

In this paper, we will discuss this approach to time-series modeling. We will first give an overview of the approach, including its motivation in the context of several applications, a brief historical perspective on the use of difference operators in numerical analysis and signal representation, and a look at the use of this approach in the *Kalman-Bucy* filtering formulation. The remainder of the paper will then focus on recent results involving the application of this approach within the aforementioned *Levinson* framework of finite linear stochastic modeling. It will be seen that, despite the loss of Toeplitz structure resulting from the transformation to divided-difference representation, recursive algorithms of complexity comparable to that of the well-known *Levinson-Durbin* algorithm (which exploits the Toeplitz structure of the shift-operator representation) can still be used to fit an appropriate incremental-difference based model. Other aspects of this problem, including a *Schur*-type algorithm and a lattice structure based on the divided-difference formulation, will also be discussed briefly.

2. Motivation & Background

Conventional statistical signal processing algorithms suffer from several disadvantages when applied to sampled data taken at high sampling rates relative to the dynamics of the underlying signals being processed. Some of these disadvantages are numerical. For example,

*This research was supported in part by the National Science Foundation under Grant CDA-91-21709, and in part by the Office of Naval Research under Grant N00014-89-J-1321.

sampled-data statistical signal processing problems become ill-conditioned at high sampling rates. This is because most such algorithms involve the inversion of the covariance matrix of the observed data. As the sampling rate increases (for fixed continuous-time bandwidth) this covariance matrix becomes increasingly stiff; and therefore numerical conditioning becomes a serious problem. A related difficulty occurs in the representation of sampled data by linear state-space models. As the amount of real time represented by the discrete time index decreases, the eigenvalues of the state matrix in such a representation will cluster about unity. If such dynamics are to be represented with finite-precision arithmetic, the preponderance of accuracy will then be devoted to the representation of the near-unity eigenvalues of such a system; whereas the interesting dynamics will be relegated to the higher-order bits in the representation. Such considerations are also manifested in several aesthetic problems. It is widely recognized that many discrete-time algorithms are formally dissimilar to their continuous-time counterparts. For example, the continuous and discrete Kalman-Bucy filters are different. Moreover, discrete-time quantities do not converge smoothly to their continuous-time counterparts. Again, the Kalman-Bucy filter is an example. These various problems are, of course, interrelated; and as we shall see here, they are manifestations of the representation of discrete time dynamics using a basic dynamical operator (the forward time shift) that does not well-approximate the basic continuous-time dynamical operator (the time derivative).

When faced with the above difficulties, one might ask: why not simply decimate (i.e., downsample) the sampled data, and avoid such problems? Certainly, for bandlimited signals, one would expect to be able to operate at the Nyquist sampling rate without undue difficulties. However, since real signals are never truly bandlimited, such decimation is usually associated with a performance penalty. More significantly though, there are several important applications in which high-speed sampling is unavoidable. For example, wideband communications formats (i.e., spread-spectrum) are central to most of the key emerging wireless telecommunications services, such as mobile radio, personal communications, and indoor wireless. In such formats, sampling rates are dictated by the radio-frequency bandwidth, which is considerably higher (often by a factor of several hundred) than the bandwidth of the underlying data signals of interest [12]. Similarly, in high-definition imaging and video, the images are oversampled both in time and in space for the purpose of providing much higher visual quality. If it is necessary to downsample in order to pro-

cess (e.g., compress) such signals, then the *raison d'être* of the high-definition format is destroyed. In the somewhat different context of digital feedback control, it is well known that sampling at system Nyquist rates can produce closed-loop instability due to the resulting time delay. In order to avoid such instability, sampling rates of 10 times bandwidth (or higher) are usually recommended [1].

In this paper, we consider an approach to such problems based on a fundamental reformulation of discrete-time dynamics, as opposed to the conventional approach of decimation. In particular, we reformulate discrete time in terms of a divided-difference operator instead of the conventional forward shift (q) operator. That is, we replace the basic dynamical operation

$$qY_k = Y_{k+1} \quad (2.1)$$

on a sequence $\{Y_k\}$, with the alternative

$$\delta Y_k = \frac{Y_{k+1} - Y_k}{\Delta} \quad (2.2)$$

where Δ is the *sampling interval*. So the operator δ is related to the operator q via

$$\delta = \frac{q - 1}{\Delta} \quad (2.3)$$

It should be noted that q and δ describe (theoretically) equivalent models, since there is a one-to-one mapping relating the two. However, the *delta* operator, which is a numerical derivative, has the advantage that it approximates the time-derivative, which is the fundamental dynamical operation of continuous time. As we shall see below, this difference is key to solving the aforementioned numerical and aesthetic problems associated with the conventional shift operator. In particular, in a finite-precision computing environment, representations in terms of the delta operator will lead to preferable algorithms for statistical signal processing. Of course, the shift-operator representation is central to a good deal of the development of fast and efficient algorithms for signal processing, so a key challenge in using an alternative operator such as δ is to find similar algorithms that are computationally competitive with their shift-operator counterparts. This is the principal issue of interest this paper.

Before continuing, it is worthwhile to take a moment to discuss the choice of the operation $\delta = \frac{q - 1}{\Delta}$ over other commonly used numerical derivatives, such as

$$\frac{1 - q^{-1}}{\Delta}; \quad \frac{2(q - 1)}{\Delta(q + 1)}; \quad \frac{q - q^{-1}}{2\Delta} \quad (2.4)$$

Although use of any of these operations to represent signal dynamics would alleviate the numerical difficulties inherent in fast sampling, the operation δ has several practical advantages from the viewpoint of algorithm development, implementation and interpretation. For example, δ has the advantage of flexibility, since it is easily transformed to q . Moreover, as $\Delta \rightarrow 0^+$, δ is a natural forward differential, which mimics the forward-differential representations prevalent in stochastic calculus. And finally, from an implementation point of view, it should be noted that dynamical equations such as

$$\delta Y = AY + Bu$$

are in "direct" form for implementation.

Of course, the use of incremental difference operators in discrete approximation problems is not a new idea. Such use dates to the early Seventeenth Century, and has its origins in the problem of subtabulation in tables of logarithms. The early formal study of incremental differences was intertwined with that of shift-operator representations from the Seventeenth through the Nineteenth Centuries, and involved such well-known figures as Newton, Stirling, Lagrange, Laplace and Cauchy, among others. (A brief account of this history is found in [4].) In the early Twentieth Century, the shift operator became central to the development of discrete-time filtering, control, and time-series analysis; whereas, the difference operators became a mainstay of numerical analysis. In more recent times, difference operators have found their way back into system and signal analysis problems, including stability theory, low-sensitivity digital filters, and improved finite wordlength digital control and state estimation.

As an illustration of this more recent work, it is of interest to consider briefly the problem of Kalman-Bucy filtering (see, e.g., [14]). In continuous time, the Kalman-Bucy problem involves a linear stochastic model of the form:

$$dY(t) = AY(t) + dV(t), \quad t \geq 0 \quad (2.5a)$$

$$dZ(t) = CY(t) + dW(t), \quad t \geq 0 \quad (2.5b)$$

where $\{Y(t); t \geq 0\}$ is a vector state process; $\{Z(t); t \geq 0\}$ is a vector measurement process; $\{V(t); t \geq 0\}$ and $\{W(t); t \geq 0\}$ are independent vector Brownian motions; and A and C are matrices of appropriate dimensions. The Kalman-Bucy filter is a recursive (in t) implementation of the conditional-mean estimator of the state $Y(t)$ based on measurements $\{Z(s); 0 \leq s \leq t\}$. The conventional sampled-data model used to derive a discrete-time version of this filter is

$$qY_k = e^{A\Delta}Y_k + V_k, \quad (2.6)$$

where, as before, Δ denotes the sampling interval, and where e denotes the matrix exponential. (Here, the subscripted variables represent discrete-time versions of the corresponding continuous-time processes.) The corresponding optimum filter involves the propagation of the dynamics described in (2.6) as part of the filtering operation. Note that these dynamics are described here by the state matrix $e^{A\Delta}$, which approaches an identity as Δ decreases towards zero. Thus, for small Δ , a conventional finite-precision representation of these dynamics will place most of its accuracy in the unit diagonal terms, and the true dynamics (represented by the higher-order term ΔA) will be relegated to the higher-order bits of the representation.

If we represent the same recursion in terms of the incremental difference δ , we have instead the representation

$$\delta Y_k = \left(\frac{e^{A\Delta} - I}{\Delta} \right) Y_k + \frac{V_k}{\Delta}. \quad (2.7)$$

Note that the state matrix now captures the dynamics as $\Delta \rightarrow 0$. A recursive filter for state estimation in such a model has been derived by Salgado, *et al.* in [14], and it is seen there that this reorganization of the dynamics improves the numerical conditioning of the filtering equations and allows their smooth convergence to continuous equations as $\Delta \rightarrow 0$.

3. The High-speed Levinson Problem

We now turn to the problem of fast algorithms for time-series modeling within the framework discussed in the preceding section. As a paradigm for this problem, we consider the autoregressive (AR) modeling problem, also known as the Levinson problem. This problem is, of course, a fundamental one in the modeling of discrete time series, and it is rich in algorithmic structure. Thus, it is a natural problem to consider in this context.

3.1. The Classical Levinson Problem

The classical Levinson autoregressive modeling problem is concerned with choosing a parameter vector $\underline{a}_n = [a_{n,0}, a_{n,1}, \dots, a_{n,n}]^T$ with $a_{n,0} \equiv 1$ to minimize the mean-squared prediction error $E\{\epsilon_n^2(t)\}$ where $\epsilon_n(t)$ is the error in the model

$$Y_{t+1} + \sum_{k=1}^n a_{n,k} Y_{t+1-k} = \epsilon_n(t), \quad t \in \mathcal{Z}, \quad (3.1)$$

and $\{Y_k\}_{k=-\infty}^{\infty}$ is an observed wide-sense-stationary (w.s.s.) random sequence.

As is well known, the coefficients minimizing this mean-squared error are the solutions to the so-called

Yule-Walker equations:

$$\mathbf{R}_n \mathbf{a}_n = \begin{pmatrix} \pi_n \\ 0 \\ \vdots \\ 0 \end{pmatrix}, \quad (3.2)$$

where \mathbf{R}_n is the $(n+1) \times (n+1)$ Toeplitz matrix whose i, j^{th} element is $c_{|i-j|}$, the $|i-j|$ -lag correlation coefficient of the signal. The *Levinson-Durbin algorithm* is an algorithm for recursively solving the Yule-Walker equations of successively higher order. This algorithm is given by [5]:

$$\mathbf{a}_{n+1} = \begin{bmatrix} \mathbf{I}_{n+1} \\ 0 \dots 0 \end{bmatrix} \mathbf{a}_n - \gamma_{n+1} \begin{bmatrix} 0 \dots 0 \\ \mathbf{J}_{n+1} \end{bmatrix} \mathbf{a}_n, \quad (3.3a)$$

for $n = 0, 1, 2, \dots$, with initialization $\mathbf{a}_0 = 1$, where, for each positive integer k , \mathbf{I}_k denotes the $k \times k$ identity matrix and \mathbf{J}_k denotes the $k \times k$ matrix that has all zero entries except for 1's in its anti-diagonal. The *reflection coefficients* $\{\gamma_n\}$ are given by

$$\gamma_{n+1} = \alpha_n / \pi_n \quad (3.3b)$$

where

$$\alpha_n = [0, \dots, 0, 1] \mathbf{R}_{n+1} \begin{bmatrix} \mathbf{a}_n \\ 0 \end{bmatrix}, \quad (3.3c)$$

and

$$\pi_n = E\{\epsilon_n^2(t)\} = [1, 0, \dots, 0] \mathbf{R}_n \mathbf{a}_n. \quad (3.3d)$$

The mean-squared error sequence $\{\pi_n\}$ satisfies the recursion

$$\pi_{n+1} = \pi_n - \alpha_n^2 / \pi_n, \quad n = 0, 1, \dots \quad (3.3e)$$

with initialization $\pi_0 = c_0$.

The numerical stability of the Levinson-Durbin algorithm for solving (3.2) has been established by Cybenko in [2]. However, as pointed out in [2], in many cases of practical interest the matrix \mathbf{R}_n is ill-conditioned, which results in unacceptable errors when the algorithm is implemented. This ill-conditioning occurs when the prediction error π_n is very small, or, equivalently, when the reflection coefficients are close to ± 1 . An important case where ill-conditioning of this nature occurs is when the discrete-time signal of interest is obtained by sampling a continuous-time process $\{Y(t); t \in \mathcal{R}\}$ at fairly rapid rates. In this case, the matrix \mathbf{R}_n is very stiff since $\lim_{\Delta \rightarrow 0} c_{|k-l|} = c_0$ for all k and l , if the underlying continuous-time process is sufficiently smooth (i.e., mean-square continuous). (Here, as before, Δ denotes

the sampling interval used to produce the discrete-time signal under study.) Since the problem is due to the poor conditioning of \mathbf{R}_n , it cannot be solved by using alternative algorithms, like the Schur algorithm, to solve (3.2), as noted in [2] and by Yagle and Levy in [18].

Moreover, in this sampled-data case, the following result can be proved.

Proposition 3.1 (Vijayan, *et al.* [17]) Assume that the continuous-time process satisfies the following conditions:

- i.) $\{Y(t); t \in \mathcal{R}\}$ has $n-1$ mean square derivatives.
- ii.) The random vector of derivatives $[Y^{(n-1)}(0), \dots, Y^{(1)}(0), Y(0)]$ has a non-singular covariance matrix.

Then,

$$\lim_{\Delta \rightarrow 0} a_{n,j} = (-1)^j \binom{n}{j}, \quad j = 0, 1, \dots, n, \quad (3.4)$$

and

$$\lim_{\Delta \rightarrow 0} \gamma_n = (-1)^{n-1}. \quad (3.5)$$

Thus we see that, as $\Delta \rightarrow 0$, if the continuous-time process has sufficiently many mean-square derivatives, the coefficients obtained by the Levinson algorithm will converge to the binomial coefficients $(-1)^j \binom{n}{j}$ independently of the underlying process. This points to a major difficulty with the standard Levinson formulation for finite-length linear modeling; namely, the parameters of this model contain no information about the statistics of the underlying process except in terms that are of higher-order in Δ .

3.2. The High-speed Levinson Problem

In order to correct the difficulties noted above, we will consider the reformulation of the autoregressive modeling problem (3.1) in terms of the divided-difference operator (2.3). To do so, we assume henceforth that the signal $\{Y_k\}_{k=-\infty}^{\infty}$ is obtained by uniformly sampling a continuous time process $\{Y(t); t \in \mathcal{R}\}$ at interval Δ .

Note that the n^{th} -order autoregressive model (3.1) can be rewritten in terms of the shift operator (2.1) as

$$A_n(q)Y_{t+1-n} = \epsilon_n(t), \quad t \in \mathcal{Z}, \quad (3.7)$$

with $A_n(q) = \sum_{k=1}^n a_{n,k} q^{n-k}$. (Here, of course, q^ℓ denotes ℓ repeated applications of q ; i.e., $q^\ell Y_k = Y_{k+\ell}$.) In

this context, it is of interest to consider an alternative model of the form

$$\left(\sum_{k=0}^n \beta_{n,k} \delta^{n-k} \right) Y_{t+1-n} = \nu_n(t), \quad t \in \mathcal{Z} \quad (3.7)$$

where $\beta_n = [\beta_{n,0}, \beta_{n,1}, \dots, \beta_{n,n}]^T$ with $\beta_{n,0} \equiv 1$, and $\{\nu_n(t)\}_{t=-\infty}^{\infty}$ is the sequence of modeling errors in this n^{th} order model. Note that the iteration of the delta operator is

$$\delta^l Y_k = \frac{\delta^{l-1} Y_{k+1} - \delta^{l-1} Y_k}{\Delta}; \quad (3.8)$$

so that

$$\delta^2 Y_k = \frac{Y_{k+2} - 2Y_{k+1} + Y_k}{\Delta^2}, \quad (3.9)$$

and so forth.

One motivation for considering the model (3.7) is its parallelism with the continuous time autoregressive model given by

$$dY^{(n-1)}(t) + \hat{a}_{n,1} Y^{(n-1)}(t) dt + \dots + \hat{a}_{n,n} Y(t) dt = dW(t)$$

where $\{W(t); t \in \mathcal{R}\}$ is a Wiener process. Another continuous-time model that has been used in [3], [7] and [8] is the following, which is based on an integral operator:

$$dY(t) + \int_{t-T}^t a(T; T - (t-s)) dY(s) = dW(s). \quad (3.10)$$

In [3], this model is approximated by using the standard discrete-time AR model with order $n = T/\Delta$. As $\Delta \rightarrow 0$, $n \rightarrow \infty$ and the limiting values of the discrete AR parameters $a_{n,j}$ are related to the continuous AR function $a(T; t)$. The disadvantage of this approach is that, for small Δ , the number of parameters in the model becomes very large (the number n of parameters should grow at a Δ^{-1} rate). In comparison, (3.7) gives a parsimonious parametrization that also converges to a continuous-time model.

Note that the variables $\delta^n Y_k, \delta^{n-1} Y_k, \dots, Y_k$ are obtained by linear transformation of the variables $q^n Y_k, q^{n-1} Y_k, \dots, Y_k$. Since $\delta^k = (q-1)^k / \Delta^k$, this transformation can be represented by T_n , an $(n+1) \times (n+1)$ matrix whose l, k^{th} element is given by

$$(T_n)_{l,k} = \frac{(-1)^{l-k}}{\Delta^{n-k}} \binom{n-k}{l-k}, \quad 0 \leq l, k \leq n. \quad (3.11)$$

(Here, we follow the convention that the binomial coefficient $\binom{n}{k} = 0$ for $k < 0$ and $k > n$.) Thus, T_n is an invertible lower triangular matrix whose $n \times n$ right

lower submatrix is T_{n-1} , with $T_0 = 1$. The inverse of this matrix is given by

$$(T_n^{-1})_{l,k} = \Delta^{n-l} \binom{n-k}{l-k}, \quad 0 \leq l, k \leq n. \quad (3.12)$$

It is straightforward to see that the vector $\underline{\beta}_n$ that solves

$$\min_{\underline{\beta}_n} E\{\nu_n^2(t)\} = ! \quad (3.13)$$

is given by

$$\underline{\beta}_n = (T_n)_{0,0} T_n^{-1} \underline{a}_n = \Delta^{-n} T_n^{-1} \underline{a}_n, \quad (3.14)$$

where \underline{a}_n solves the Yule-Walker equations (3.2). So the parametrization (3.7) is simply a linear transformation of the conventional one (3.1); i.e., both describe the essentially same linear relationship among the same set of random variables. Of course, the key difference between the two descriptions is in how they represent the coefficients in this relationship.

It is the Toeplitz property of R_n , the n^{th} -order covariance matrix of the signal, that makes it possible to solve (3.2) recursively using $O(n^2)$ computations via (3.3). Since T_n is triangular, if we knew R_n , it would be possible to solve (3.13) using $O(n^2)$ computations, by first solving (3.2) for \underline{a}_n using the Levinson-Durbin algorithm, and then using (3.14) to obtain $\underline{\beta}_n$. However, in this procedure, any numerical errors in calculating \underline{a}_n due to the ill-conditioning of R_n would carry over to the calculation of $\underline{\beta}_n$. Also, this type of calculation is not recursive in n .

Exploiting the special structure of the matrix T_n , Vijayan, *et al.* [17] have obtained an $O(n^2)$ algorithm for solving (3.13), that only requires knowledge of the non-Toeplitz covariance matrix of $(\delta^n Y_k, \delta^{n-1} Y_k, \dots, Y_k)$. This algorithm has the added advantage of being recursive, like the Levinson algorithm. It is summarized in the following:

Proposition 3.2 (Vijayan, *et al.* [17]) The argument solving (3.16) is given recursively (in n) by

$$\underline{\beta}_{n+1} = C_n \underline{\beta}_n + \frac{1}{\Delta^2} \frac{\tilde{\gamma}_{n+1}}{\tilde{\gamma}_n} \begin{bmatrix} 0 \\ \underline{\beta}_n \end{bmatrix} - \frac{\tilde{\alpha}_n}{\tilde{\alpha}_{n-1}} \begin{bmatrix} 0 \dots 0 \\ C_{n-1} \end{bmatrix} \underline{\beta}_{n-1} \quad (3.15a)$$

for $n = 0, 1, \dots$, with initialization $\underline{\beta}_{-1} = 0$, $\underline{\beta}_0 = 1$, and $\tilde{\gamma}_0 = -\Delta^{-2}$. Here C_n is the $(n+2) \times (n+1)$ matrix

defined by

$$C_n = \frac{1}{\Delta} \begin{bmatrix} \Delta & 0 & 0 & \dots & 0 \\ 1 & \Delta & 0 & \dots & 0 \\ 0 & 1 & \Delta & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & 1 & \Delta \\ 0 & 0 & \dots & 0 & 1 \end{bmatrix}; \quad (3.15b)$$

$\tilde{\gamma}_n$ is defined by

$$\tilde{\gamma}_{n+1} = \tilde{\alpha}_n / \tilde{\pi}_n, \quad n = 1, 2, \dots, \quad (3.15c)$$

with

$$\tilde{\alpha}_n = [0, \dots, 0, 1] Q_{n+1} \begin{bmatrix} \beta_n \\ 0 \end{bmatrix}, \quad (3.15d)$$

for $n = 1, 2, \dots$, and

$$\tilde{\alpha}_0 = E\{x(t)\delta x(t)\} + E\{x^2(t)\} / \Delta; \quad (3.15e)$$

and

$$\tilde{\pi}_n = E\{\nu_n^2(t)\} = [1, 0, \dots, 0] Q_n \beta_n, \quad (3.15f)$$

for $n = 0, 1, 2, \dots$, with Q_n the covariance matrix of $(\delta^n Y_k, \dots, \delta Y_k, Y_k)$. (Q_n does not depend on k due to the assumed stationarity of the signal.)

The algorithm (3.15) can be derived directly by substitution of (3.14) into (3.3) and then making use of the combinatorial structure of the matrices T_n . The matrices Q_n also have special structure that make them amenable to fast inversion within the more general theory of Heinig and Rost [6]. In particular, they are *Toeplitz like*. This structure will be discussed further below.

In comparing (3.15) with its Levinson counterpart (3.3), it is straightforward to see that the complexity of the new algorithm is of the same order despite the fact that the algorithm inverts a non-Toeplitz matrix. It should be noted, however, that (3.15) does involve more operations than (3.3); but, the difference in the two complexities is a constant scale factor. However, unlike the Levinson coefficients, it can be shown that (under sufficient smoothness) the delta-Levinson coefficients converge to meaningful statistical parameters of the underlying process - namely, the regression coefficients of the n^{th} mean-square derivative on those of lower orders. Moreover, in the limit (3.15) has the form

$$\beta_{n,j} = \beta_{n-1,j} + \frac{\tilde{\pi}_{n-1,0}}{\tilde{\pi}_{n-2,0}} \beta_{n-2,j-2}, \quad (3.16)$$

for $j = 0, \dots, n$, which is the same as a Levinson type recursion for continuous-time autoregressive models derived by Pham and le Breton in [10]. Thus, the recursion itself takes on a meaningful (and stable) limiting form.

These favorable limiting properties suggest that the delta-Levinson formulation will be more stable numerically for small Δ than is the standard Levinson formulation. Numerical results using floating-point calculations reported in [17] support this supposition; and, in fact, the numerical performance of (3.15) for high sampling rates is considerably better than that of the classical Levinson algorithm. (It should be noted that alternative versions of the Levinson-Durbin algorithm have been derived, for example, by le Roux and Gueguen in [9], that are also more robust to finite precision effects. However, the divided-difference formulation provides a formal, generalizable approach to this problem.)

This superior numerical stability of the delta model can be explained in terms of the limiting results discussed above. In particular, there is a one-to-one correspondence between the parameter vectors \underline{a}_n and $\underline{\beta}_n$. However, the limiting value of \underline{a}_n is independent of the statistics of the process. Hence, the useful information in the parameter vector is being "compressed". As a result of this, small perturbations in the coefficients can cause large variations in the modeling error. The delta coefficients do not suffer from this problem since their limiting values contain useful information about the process.

4. Discussion

We see from the previous section that the delta version of the Levinson problem offers a number of advantages over the standard one for high-speed processing. However, the Levinson formulation has several useful properties that are not obviously present in its delta counterpart. Such issues as lattice implementation for layered adaptivity [5], Schur realization for parallelizability [18], and direct forms for calculating reflection coefficients with covariance data [13], fall within this category of problems. Other interesting issues include estimation techniques for extracting the covariance matrix Q_n from noisy data (a process almost certainly to require regularization).

Progress has been made on some of these issues within the delta context. For example, in [15] a parallelizable Schur-type delta algorithm has been developed by exploiting the Toeplitz-like structure of Q_n , defined in the sense of Heinig and Rost [6]. This algorithm exhibits the parallelizability of the conventional Schur algorithm, while retaining the numerical advantages of the delta-Levinson algorithm. Lattice structures based on the divided-differences have also been developed recently, as

described in [16]. Unlike the Schur algorithm, however, these lattices do not enjoy all of the favorable properties of the conventional shift-operator lattice. For example, the shift-operator lattice can be adapted in layers, since the first p stages of the lattice depend only on the first p lag covariances. However, this is not the case for the delta-based lattices described in [15]. The development of a layered lattice based on the delta operator is an open problem.

Although the Levinson problem is perhaps the most fundamental of time-series modeling problems, it is of course only one of many such problems. Whether the methods described herein are suitable for other modeling problems is largely an open question, although progress has been made in the ARMA identification problem [4]. However, in view of the success of these methods in the Levinson problem, they offer a promising methodology for application in other time-series modeling problems involving rapidly sampled data.

5. References

1. Åström, K. J. and Wittenmark, B. (1984), *Computer Controlled Systems: Theory and Design*, Englewood Cliffs, NJ: Prentice-Hall.
2. Cybenko, G. (1980), "The numerical stability of the Levinson-Durbin algorithm for Toeplitz systems of equations," *SIAM J. Scient. Statist. Comp.*, 1, 303-319.
3. Dewilde, P., Vieira, A. C. and Kailath, T. (1978), "On a generalized Szegő-Levinson realization algorithm for optimal linear predictors based on a network synthesis approach," *IEEE Trans. Circuits Syst.*, CAS-25, 663-675.
4. Goodwin, G. C., Middleton, R. H. and Poor, H. V. (1992), "High-speed digital signal processing and control," *Proc. IEEE*, 80, 240-259.
5. Goodwin, G. C. and Sin, S. (1984), *Adaptive Filtering, Prediction and Control*, Englewood Cliffs, NJ: Prentice-Hall.
6. Heinig, G. and Rost, K. (1988), *Algebraic Methods for Toeplitz-like Matrices and Operators*, Berlin: Springer-Verlag.
7. Kailath, T., Levy, B. C., Ljung, L. and Morf, M. (1978), "Fast time-invariant implementations of Gaussian signal detectors," *IEEE Trans. Inform. Theory*, IT-24, 469-477.
8. Kailath, T., Vieira, A. and Morf, M. (1978), "Inverses of Toeplitz operators, innovations, and orthogonal polynomials," *SIAM Rev.*, 20, 106-119.
9. le Roux, J. and Gueguen, C. (1977), "A fixed point computation of partial correlation coefficients," *IEEE Trans. Acoust., Speech, and Signal Processing*, ASSP-25, 257-259.
10. Pham, D. T. and le Breton, A. (1991), "Levinson Durbin type algorithms for continuous time autoregressive models and applications," *Mathematics of Control, Signals, and Systems*, 4, 69-79.
11. Poor, H. V. (1988), *An Introduction to Signal Detection and Estimation*, New York: Springer-Verlag.
12. Poor, H. V. (1992), "Signal processing for wide-band communications," *IEEE Information Theory Society Newsletter*, 42 (2), 1-10.
13. Rialan, C. P. and Scharf, L. L. (1989), "Fixed-point error analysis of the lattice and the Schur algorithms for the autocorrelation method of linear prediction," *IEEE Trans. Acoust., Speech, and Signal Processing*, 37, 1950-1957.
14. Salgado, M., Middleton, R. H. and Goodwin, G. C. (1988), "Connection between continuous and discrete Riccati equations with applications to Kalman filtering," *IEE Proceedings-D*, 135, 28-34.
15. Vijayan, R. and Poor, H. V. (1992), "Fast triangular factorization of covariance matrices of differenced time series," *SIAM J. Matrix Anal. Appl.*, to appear.
16. Vijayan, R. and Poor, H. V. (1992), "Lattice filters based on the delta operator," submitted to *Systems and Control Letters*.
17. Vijayan, R., Poor, H. V., Moore, J. B. and Goodwin, G. C. (1991), "A Levinson-type algorithm for modeling fast-sampled data," *IEEE Trans. Automat. Contr.*, 36, 314-321.
18. Yagle, A. E. and Levy, B. C. (1985), "The Schur algorithm and its applications," *Acta Applicandae Mathematicae*, 3, 255-284.

Resampling permutations in regression with exchangeable errors

Raoul LePage* and Krzysztof Podgórski†

Department of Probability and Statistics
Michigan State University
East Lansing, MI 48824

Abstract

For linear regression with exchangeable errors we observe that random permutations of the sample residuals can *with high probability* approximately recover the joint sampling distribution of the errors of arbitrary contrasts, conditional on the order statistics of the errors actually present in the data.

1. Introduction

Let $Y = X\beta + \epsilon$ where X is an $n \times d$ finite matrix of real numbers, β is a $d \times 1$ vector of real numbers, and ϵ is an $n \times 1$ exchangeable random vector. An $n \times 1$ vector v of real numbers is called a *contrast* if the entries of v sum to zero, equivalently if the Euclidean scalar product $v \cdot 1 = 0$ where 1 denotes the vector of 1's.

Ordinarily one is interested in estimating the joint distribution for several v of $v \cdot \epsilon$ since it is equal to the estimation error, i.e. $v \cdot \epsilon = (v \cdot Y) - (v \cdot X\beta)$. For i.i.d. errors with finite variance, the central limit theorem is popularly used for this purpose. Bootstrap methods apply to this case as well [5], [6], [9] but how are we to know if the underlying assumptions have practical application to the data at hand [8]?

Instead, we propose to estimate the sampling distribution of $v \cdot \epsilon$ conditional on the sigma field \mathcal{F} generated by the order statistics of the coordinates of ϵ . We use the notation $v \cdot \epsilon | \mathcal{F}$ to denote this conditional distribution. Denote by ϵ^\perp the vector of residuals $Y - \hat{Y}$, where $\hat{Y} = Y/X$ is the projection of Y to the column space of X . Let π denote a uniformly distributed random permutation applied to the coordinates of n -space.

Suppose that the distribution of π , conditional on ϵ , is also uniform over all $n!$ permutations, and therefore that π is independent of ϵ . We will observe that, provided $(d-1)/(n-1)$ is small, $v \cdot \pi \epsilon^\perp | Y$ is *with high probability* a close approximation of $v \cdot \pi \epsilon | \mathcal{F}$. Notice that the last distribution is equal to $v \cdot \epsilon | \mathcal{F}$ since ϵ is exchangeable.

Proposal: Estimate the joint \mathcal{F} -conditional sampling distributions of contrasts $v_k \cdot \epsilon$ by the Y -conditional distributions of $v_k \cdot \pi \epsilon^\perp$, $k \leq m$.

Our proposal is related to, but fundamentally differs from, the approach taken in [7] when developing *descriptive* tests of linear hypotheses. To quote from [7], "our reference sets are derived by permuting residuals, and our significance level is a descriptive statistic rather than a probability."

2. Main Result

We assume the notation of the introduction. As before, ϵ is assumed to be exchangeable and independent of a random uniformly distributed permutation π . If u is an $n \times 1$ vector then by \bar{u}, u^2 we will denote the arithmetic average and the vector of squares of coordinates, respectively. By s_u^2 we will denote the $(n-1)$ -divisor sample variance of u i.e. $s_u^2 = n(\bar{u} - \bar{u})^2 / (n-1)$. Notice that for any contrast v both $v \cdot \pi \epsilon^\perp | Y$ and $v \cdot \pi \epsilon | \mathcal{F}$ are centered at zero since, by independence of π and ϵ and the fact that $E(\pi u) = \bar{u}1 = n^{-1}(u \cdot 1)1$ for any $n \times 1$ vector, we have

$$E(v \cdot \pi \epsilon^\perp | Y) = E(\pi v \cdot \epsilon^\perp | \epsilon) = E(\pi v) \cdot E(\epsilon^\perp | \epsilon) = 0$$

and

$$E(v \cdot \pi \epsilon | \mathcal{F}) = E(\pi v \cdot \epsilon | \mathcal{F}) = E(\pi v) \cdot E(\epsilon | \mathcal{F}) = 0.$$

In all that follows $E^{\mathcal{F}}, E^\epsilon$ will denote conditional expectation with respect to \mathcal{F} and ϵ , respectively.

*Research partially supported by ONR Grant N00014-91-J-1087.

†On leave from Hugo Steinhaus Center, Technical University of Wrocław

Proposition 1 . *If $\mathbf{1}$ is in the column space of \mathbf{X} which has rank d and $v_k, k = 1, \dots, m$ are contrast vectors, then*

$$E^{\mathcal{F}} \frac{\sum_{k=1}^m E^{\epsilon} [(v_k \cdot \pi \epsilon) - (v_k \cdot \pi \epsilon^{\perp})]^2}{\sum_{k=1}^m E^{\epsilon} (v_k \cdot \pi \epsilon)^2} = \frac{d-1}{n-1}.$$

For a contrast vector v , $\gamma(\epsilon) = E^{\epsilon} (v \cdot \pi \epsilon - v \cdot \pi \epsilon^{\perp})^2 / E^{\epsilon} (v \cdot \pi \epsilon)^2$ and denoting by F_1, F_2 the distribution functions of $v \cdot \epsilon / \sqrt{E^{\mathcal{F}} (v \cdot \epsilon)^2} | \mathcal{F}$, $v \cdot \pi \epsilon^{\perp} / \sqrt{E^{\mathcal{F}} (v \cdot \epsilon)^2} | Y$, respectively, we have the following inequalities.

Corollary 1 . *For any positive δ_1, δ_2 we have*

$$P(\gamma(\epsilon) > \sqrt{\frac{d-1}{n-1}} \delta_1) \leq \sqrt{\frac{d-1}{n-1}} \delta_1^{-1},$$

$$F_1(x) \in [F_2(x - \delta_2) - \frac{\gamma(\epsilon)}{\delta_2^2}, F_2(x + \delta_2) + \frac{\gamma(\epsilon)}{\delta_2^2}].$$

For the proofs see Section 5.

3. Example

We now present a simple example which illustrates the advantage of resampling by permutations vs other methods including traditional bootstrapping. The vector of errors considered here has one value dominating the other ones and typifies what can happen when the distribution of errors has a long tail (in particular when our errors do not satisfy the usual assumptions about existence of moments). Consider an $n \times 2$ matrix \mathbf{X} defined as follows

$$\mathbf{X} = \begin{bmatrix} 1 & 1 \\ \vdots & \vdots \\ 1 & 1 \\ 1 & -1 \\ \vdots & \vdots \\ 1 & -1 \end{bmatrix}.$$

We assume that n is even and the columns $X_1 = \mathbf{1}$ and X_2 of \mathbf{X} are orthogonal. Suppose that our actual errors consist of an n -vector ϵ given by

$$\epsilon = \begin{bmatrix} a \\ 0 \\ \vdots \\ 0 \end{bmatrix}.$$

One can notice at once that

$$\begin{aligned} \epsilon / \mathbf{X} &= \frac{\mathbf{1} \cdot \epsilon}{\|\mathbf{1}\|^2} \cdot \mathbf{1} + \frac{X_2 \cdot \epsilon}{\|X_2\|^2} \cdot X_2 \\ &= \frac{a}{n} \cdot \mathbf{1} + \frac{a}{n} \cdot X_2 = \begin{bmatrix} 2a/n \\ \vdots \\ 2a/n \\ 0 \\ \vdots \\ 0 \end{bmatrix}, \end{aligned}$$

and thus

$$\epsilon^{\perp} = \begin{bmatrix} a - 2a/n \\ -2a/n \\ \vdots \\ -2a/n \\ 0 \\ \vdots \\ 0 \end{bmatrix},$$

Let us take for our contrast vector v the second column X_2 of the matrix \mathbf{X} . One can easily observe that the distribution $v \cdot \epsilon | \mathcal{F}$ is concentrated on two points a and $-a$ with the equal weights $1/2$. This can be illustrated as follows.

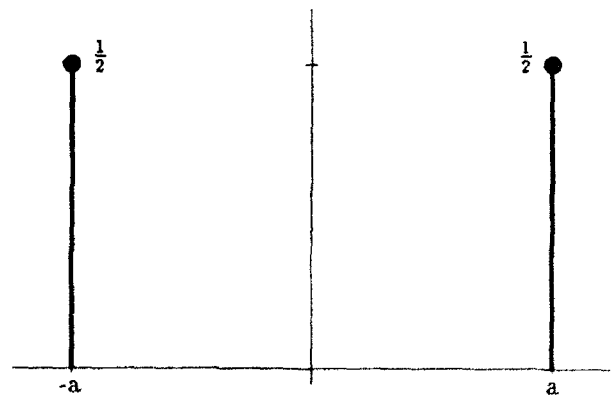


Fig.3.1. Distribution of $v \cdot \epsilon | \mathcal{F}$.

We will compare three different methods of estimating this distribution: the resampling by permutation of residuals, the traditional bootstrap with replacement and the normal approximation. The next three pictures present first order approximations of these distributions for $n = 625$. See Section 6 for details.

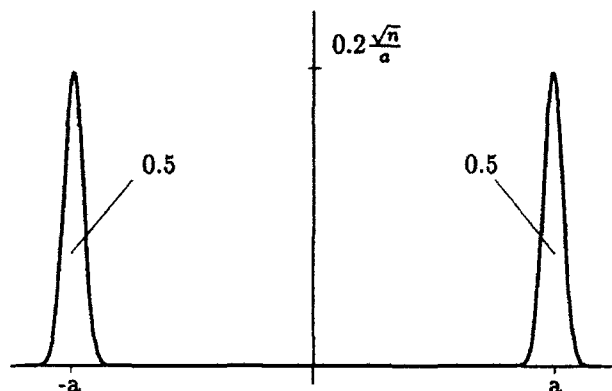


Fig.3.2. The resampling by permutation.

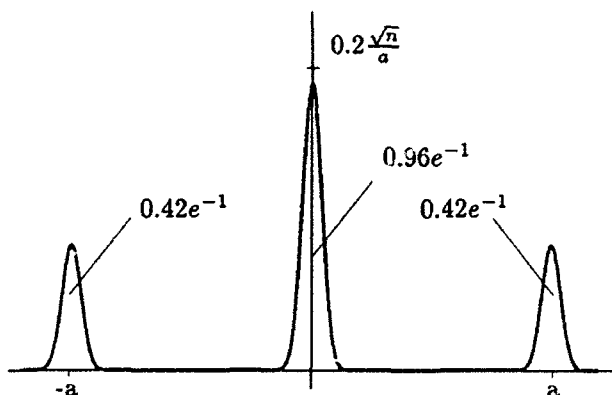


Fig.3.3. The traditional bootstrapping.

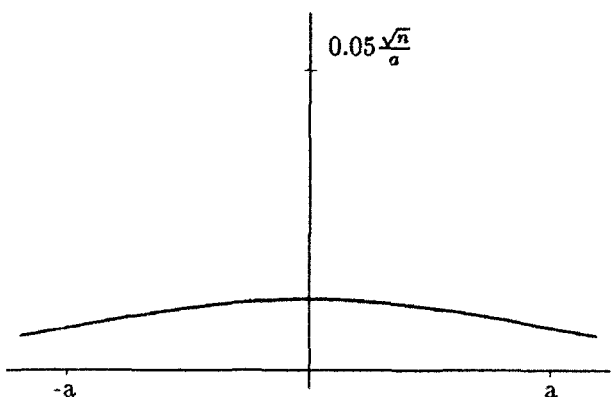


Fig.3.4. The normal approximation.

4. Computer simulation of resampling for stable errors

As a second illustration we have made computer simulations which compare resampling permutations of errors vs resampling permutations of residuals in a linear regression model fitting a cubic polynomial. Consider a 101-dimensional symmetric α -stable error ϵ with independent coordinates and a 101×4 matrix \mathbf{X} obtained by substitution of equally spaced x -values in the interval $[0, 1]$ to the vector of monomials (x^0, x^1, x^2, x^3) , so $x_{i,j} = x_i^{j-1}$ $j = 1, 2, 3, 4$, $x_i = (i-1)/101$, $i = 1, \dots, 101$. We have used the Chambers et. al. formula [3] to generate stable errors by pseudo-random numbers. Since our result does not require any assumption on moments of errors we have observed three notable cases: the normal distribution ($\alpha = 2$), Cauchy distribution ($\alpha = 1$) and $\alpha = 0.8$. We first compute three 101×1 row vectors v_i , $i = 1, 2, 3$ of the matrix $(\mathbf{X}^T \mathbf{X})^{-1} \mathbf{X}^T$ as well as the matrix of projection on the column space of \mathbf{X} . Then, by generating pseudo-random permutations π we get a sample of two hundred errors $v_i \cdot \pi \epsilon$, residuals $v_i \cdot \pi \epsilon^\perp$ and projections of errors $v_i \cdot \pi \hat{\epsilon}$, $i = 1, 2, 3$. Visual comparison of the values of these samples is made on parallel plots. Since scale changes in ϵ apply equally to $v_i \cdot \pi \epsilon$, $v_i \cdot \pi \epsilon^\perp$, $v_i \cdot \pi \hat{\epsilon}$ it is the relative comparisons between these plots which are important. Four parallel lines represent the coordinate axes of four dimensional space and any point of this space corresponds to a polygonal line which joins values of the coordinates on each axis. For convenience all polygonal lines are started at the origin. For details concerning parallel plots see [4].

In these parallel plots we see a very close agreement between the multivariate sampling distribution of $\{v_i \cdot \pi \epsilon, i = 1, 2, 3\}$ and that of $\{v_i \cdot \pi \epsilon^\perp, i = 1, 2, 3\}$ conditional on the given ϵ . As might be expected the agreement is even better than suggested by the sampling distribution of $\{v_i \cdot \pi \hat{\epsilon}, i = 1, 2, 3\}$. For this example $(d-1)/(n-1) = .03$.

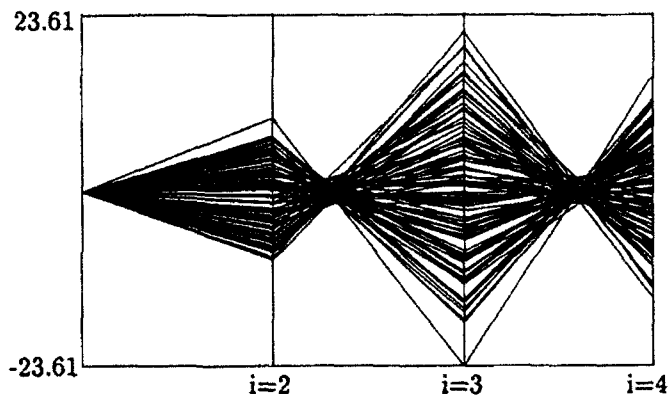


Fig.4.1a. $\alpha = 2, v_i \cdot \pi \epsilon$.

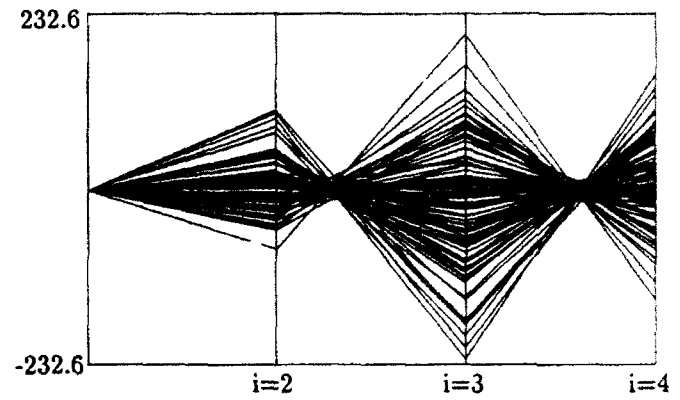


Fig.4.2a. $\alpha = 1, v_i \cdot \pi \epsilon$.

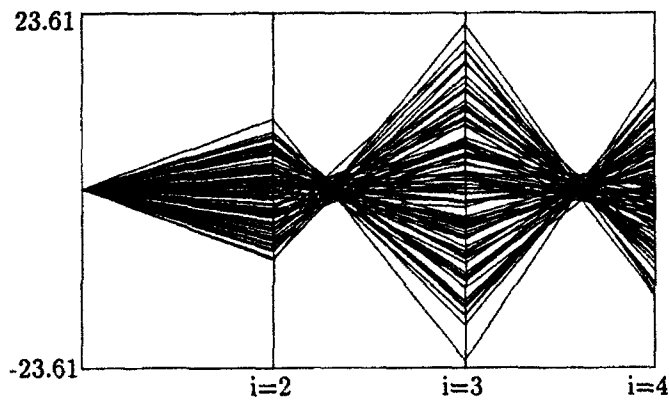


Fig.4.1b. $\alpha = 2, v_i \cdot \pi \epsilon^\perp$.

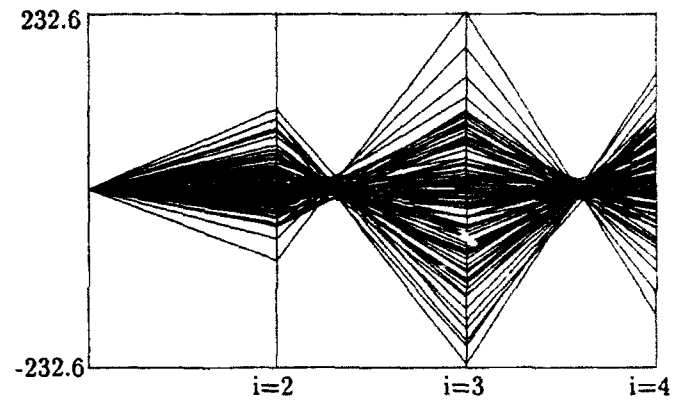


Fig.4.2b. $\alpha = 1, v_i \cdot \pi \epsilon^\perp$.

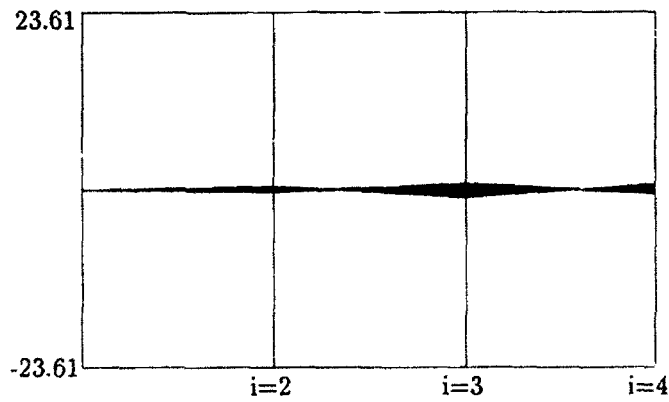


Fig.4.1c. $\alpha = 2, v_i \cdot \pi \hat{e}$.

Fig.4.1

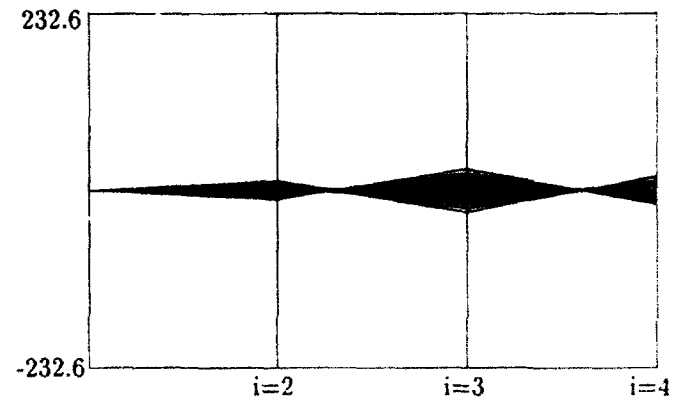


Fig.4.2c. $\alpha = 1, v_i \cdot \pi \hat{e}$.

Fig.4.2

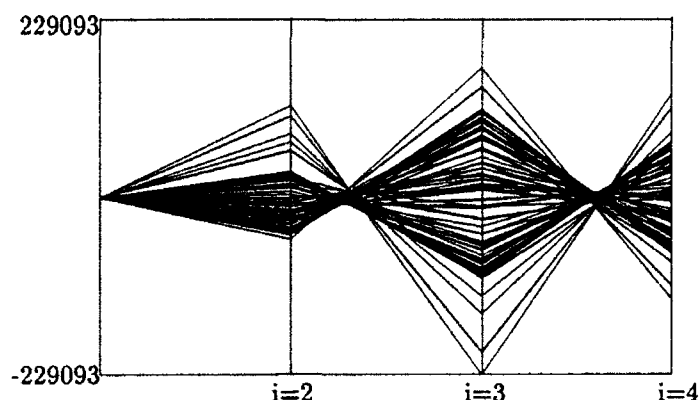
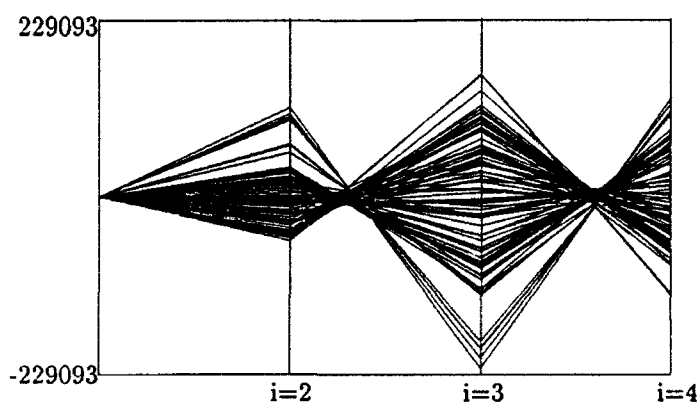
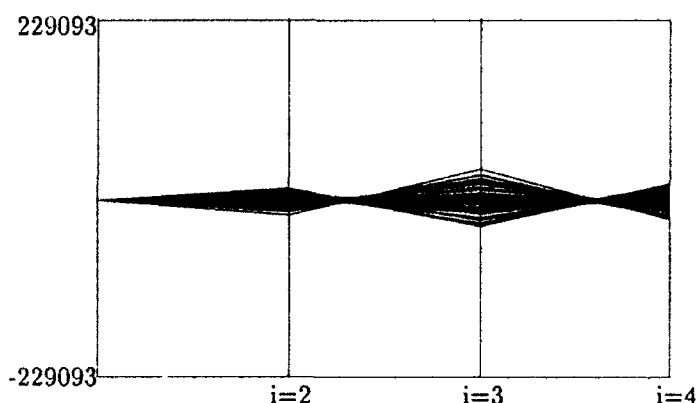
Fig.4.3a. $\alpha = 0.8, v_i \cdot \pi \epsilon$.Fig.4.3b. $\alpha = 0.8, v_i \cdot \pi \epsilon^\perp$.Fig.4.3c. $\alpha = 0.8, v_i \cdot \pi \hat{\epsilon}$.

Fig.4.3

5. Appendix I

Variants of the following two lemmas are apparently well known ([4] Section 4.1, [7], pg 297, (21)). We present their simple proofs.

Lemma 1. Let v be a contrast vector and ξ a random $n \times 1$ -vector independent of a random permutation π . Then

$$E^{\mathcal{F}}(v \cdot \pi \xi)^2 = \|v\|^2 E^{\mathcal{F}}(s_{\xi}^2).$$

Proof. We have

$$\begin{aligned} E^{\mathcal{F}}(v \cdot \pi \xi)^2 &= \frac{1}{n!} \sum_{\sigma \in \Sigma_n} E^{\mathcal{F}}(v \cdot \sigma \xi)^2 = \\ &= E^{\mathcal{F}}\left[\sum_{a=1}^n v_a^2 \xi_{\sigma(a)}^2 + \sum_{a \neq b} v_a v_b \xi_{\sigma(a)} \xi_{\sigma(b)}\right] = \\ &= E^{\mathcal{F}}\left[\left(\sum_{a=1}^n v_a^2\right) \bar{\xi}^2 + \frac{1}{n(n-1)} \left(\sum_{a \neq b} v_a v_b\right) \left(\sum_{a \neq b} \xi_a \xi_b\right)\right] = \\ &= \|v\|^2 E^{\mathcal{F}} \bar{\xi}^2 + \frac{1}{n-1} [\|v\|^2 - (v \cdot 1)] E^{\mathcal{F}}(\bar{\xi}^2 - n \bar{\xi}^2), \end{aligned}$$

where Σ_n denotes the set of all permutations of $1, \dots, n$. Since $(v \cdot 1) = 0$ we obtain

$$E^{\mathcal{F}}(v \cdot \pi \xi)^2 = \|v\|^2 \frac{n}{n-1} E^{\mathcal{F}}(\bar{\xi}^2 - \bar{\xi})^2 = \|v\|^2 E^{\mathcal{F}}(s_{\xi}^2).$$

□

Corollary 2. For any contrast vector v we have

$$E^{\mathcal{F}}(v \cdot \epsilon)^2 = \|v\|^2 s_{\epsilon}^2.$$

Proof. It follows immediately from Lemma 1 if we notice that $v \cdot \pi \epsilon|_{\mathcal{F}} = v \cdot \epsilon|_{\mathcal{F}}$ and that s_{ϵ}^2 is \mathcal{F} -measurable.

□

The next result can be interpreted to mean that the relative error caused by resampling $\pi \epsilon^\perp$ instead of $\pi \epsilon$ is on the average small provided $d-1$ is small relative to $n-1$.

Lemma 2. If 1 is in the column space of X which has rank d and v is a contrast vector, then

$$E^{\mathcal{F}} \frac{E^{\epsilon}[(v \cdot \pi \epsilon) - (v \cdot \pi \epsilon^\perp)]^2}{E^{\epsilon}(v \cdot \pi \epsilon)^2} = \frac{d-1}{n-1}.$$

Proof. Notice that by Lemma 1

$$E^\epsilon(v \cdot \pi\epsilon)^2 = E^\mathcal{F}(v \cdot \pi\epsilon)^2 = \|v\|^2 s_\epsilon^2.$$

The same lemma gives us also

$$\begin{aligned} E^\mathcal{F} E^\epsilon[(v \cdot \pi\epsilon) - (v \cdot \pi\epsilon^\perp)]^2 &= E^\mathcal{F}(v \cdot \pi\epsilon/\mathbf{X}) \\ &= \|v\|^2 E^\mathcal{F}(s_\epsilon^2/\mathbf{X}) \end{aligned}$$

and since s_ϵ^2 is \mathcal{F} -measurable it proves that

$$E^\mathcal{F} \frac{E^\epsilon[(v \cdot \pi\epsilon) - (v \cdot \pi\epsilon^\perp)]^2}{E^\epsilon(v \cdot \pi\epsilon)^2} = E^\mathcal{F} \frac{s_\epsilon^2/\mathbf{X}}{s_\epsilon^2}.$$

The proof is completed using an equality $E^\mathcal{F} s_\epsilon^2/\mathbf{X} = ((d-1)/(n-1))s_\epsilon^2$ obtained by Box and Watson ([2], Section 4.1) in their study of F -distribution approximations for $s_\epsilon^2/\mathbf{X}/s_\epsilon^2$ with non-normal errors ϵ . It may be obtained from Lemma 1 as follows

$$\begin{aligned} E^\mathcal{F} s_\epsilon^2/\mathbf{X} &= \frac{1}{n-1} E^\mathcal{F} (\|\epsilon/\mathbf{X}\|^2 - (\epsilon/\mathbf{X} \cdot \mathbf{1})^2/n) \\ &= \frac{1}{n-1} E^\mathcal{F} \left(\sum_{a=1}^d (u_a \cdot \epsilon)^2 - (\epsilon \cdot \mathbf{1})^2/n \right) \\ &= \frac{1}{n-1} \sum_{a=2}^d E^\mathcal{F} (u_a \cdot \epsilon)^2 \\ &= \frac{d-1}{n-1} s_\epsilon^2. \end{aligned}$$

□

Notice that the proof of Lemma 2 can be applied without any change to obtain Proposition 1.

Further, if we denote by \mathcal{X} the orthogonal complement of $\mathbf{1}$ in the column space of \mathbf{X} we have then

$$E^\mathcal{F} \frac{E^\epsilon \|(\pi\epsilon - \pi\epsilon^\perp)/\mathcal{X}\|^2}{E^\epsilon \|(\pi\epsilon)/\mathcal{X}\|^2} = \frac{d-1}{n-1}.$$

Indeed, for any vector u we have by the definition of s_u^2 that $s_u^2 = s_{u/\mathcal{X}}^2$ and $\|u\|^2 = (u \cdot \mathbf{1})^2/n + (n-1)s_u^2$. Thus

$$\begin{aligned} E^\mathcal{F} \|(\pi\epsilon - \pi\epsilon^\perp)/\mathcal{X}\|^2 &= (n-1) E^\mathcal{F} s_{\pi(\epsilon)/\mathbf{X}}^2 \\ &= (n-1) E^\mathcal{F} s_{\epsilon/\mathbf{X}}^2 \end{aligned}$$

and

$$E^\mathcal{F} \|(\pi\epsilon)/\mathcal{X}\|^2 = (n-1) E^\mathcal{F} s_\epsilon^2.$$

Finally, the facts included in Corollary 2.1 follow from the Markov inequality and from

$$F_1(x) \leq F_2(x + \delta) + P(Z_2 - Z_1 \geq \delta), \quad \delta > 0$$

which is true for any random variables Z_1, Z_2 with marginal distribution functions F_1, F_2 .

6. Appendix II

We present some facts about exact distributions and their approximations which were described and illustrated in Section 3.

We propose $v \cdot \pi\epsilon^\perp/\epsilon$ as a good approximation of $v \cdot \pi\epsilon/\mathcal{F}$. Notice first that this random variable has the same distribution as $\pi v \cdot \epsilon^\perp/\epsilon$ and thus takes the values

$$\begin{aligned} y_k^\pm &= \pm(a - 2a/n - 2ak/n + 2a(l - k - 1)/n) \\ &= \pm 2a(1 - 2(k+1)/n) \end{aligned}$$

with corresponding probabilities

$$\frac{\binom{l-1}{k} \binom{l}{l-1-k}}{2 \binom{n-1}{l-1}},$$

where $k = 0, \dots, l-1$ and $l \equiv n/2$. This follows from the fact that it corresponds to the distribution which samples one's or minus one's. The above probability is assigned to the event when the choice consists of k one's (minus one's) and $l-1-k$ minus one's (one's). The weight $1/2$ comes from the two symmetric events in which the first value of πv is equal either to one or to minus one.

Let γ denote the hypergeometric distribution with parameters $n-1, l-1, l-1$. Then around points $\pm a$ our conditional distribution corresponds to the distribution of the random variable $\pm \xi = \pm 2a(1 - 2(\gamma+1)/n)$. Since

$$E(\gamma) = (l-1)^2/(n-1) \approx l/2 = n/4,$$

and

$$Var(\gamma) = \frac{(l-1)^2 l^2}{(n-1)^2 (n-2)} \approx n/16,$$

for large n we have approximately

$$E(\pm \xi) \approx \pm(a - 4a/n) \approx \pm a,$$

and

$$\begin{aligned} \text{Var}(\xi) &\approx E(-4a/n(\gamma - n/4) - 4a/n)^2 = \\ &= 16a^2/n^2 E(\gamma - n/4)^2 + 16a^2/n^2 \\ &\approx a^2/n(1 + 16/n) \approx a^2/n. \end{aligned}$$

This confirms the fact that our distribution is concentrated pretty close to points a and $-a$ and thus is reasonable approximation of the distribution of interest. One can also argue through the normal approximation of γ for large n . Namely, it follows from Theorem 3 of [1] that $(4\gamma - n)/\sqrt{n}$ converges in law to the standard normal distribution $\mathcal{N}(0, 1)$. So for large n distribution of γ is approximately equal to $\mathcal{N}(n/4, n/16)$ and consequently the distribution of ξ is equal to $\mathcal{N}(a, a^2/n)$.

Now consider the traditional bootstrap where the distribution of interest is approximated by the distribution of $v \cdot (\epsilon^\perp)^* | \epsilon$, where $(\epsilon^\perp)^*$ indicates random variable obtained by sampling n -times with replacement from values of the vector ϵ^\perp . This random variable is distributed over points of the form

$$(k_1 - k_2)(a - 2a/n) + 2a(l_2 - l_1)/n$$

with probabilities

$$T_{l, \frac{1}{n}, \frac{1}{2} - \frac{1}{n}}(k_1, l_1) T_{l, \frac{1}{n}, \frac{1}{2} - \frac{1}{n}}(k_2, l_2),$$

where $k_1, k_2 = 0, \dots, n$, $l_1 = 0, \dots, l - k_1$, $l_2 = 0, \dots, l - k_2$. Here and henceforth T_{l, p_1, p_2} stands for the trinomial distribution given by

$$T_{l, p_1, p_2}(x, y) = \binom{l}{x} \binom{l-x}{y} (p_1)^x (p_2)^y (p_3)^{l-x-y},$$

where $x = 0, \dots, l$, $y = 0, \dots, l - x$ and $p_3 = 1 - p_1 - p_2$. Similarly, $B_{n, p}$ will denote the binomial distribution. Now one can notice that the distribution of $v \cdot (\epsilon^\perp)^* | \epsilon$ has asymptotically nonvanishing mass around zero.

Indeed, from the following equality

$$T_{l, p_1, p_2}(x, y) = B_{l, p_1}(x) B_{l-x, \frac{p_2}{1-p_1}}(y),$$

and taking $k_1 = k_2 = 0$ we obtain that the points $2a(l_2 - l_1)/n$, $l_1, l_2 = 0, \dots, l$ are distributed with probabilities

$$(1 - 1/n)^n B_{l, \frac{n-2}{2n-2}}(l_1) B_{l, \frac{n-2}{2n-2}}(l_2).$$

and using the Poisson approximation for the binomial distribution and Central Limit Theorem we will obtain for large n the approximate distribution equal to

$e^{-1} \mathcal{N}(0, a^2/n)$, which gives asymptotically the mass equal to e^{-1} . In fact the mass will be even bigger since we should take into account not only $k_1 = k_2 = 0$ but also all cases when $k_1 - k_2 = 0$. This implies that in this case the traditional bootstrap fails to recover the distribution $v \cdot \epsilon | \mathcal{F}$.

It is possible to explore the distribution of $v \cdot (\epsilon^\perp)^* | \epsilon$ even more carefully since it is equal to the distribution of $\tilde{\gamma}^s \cdot \begin{bmatrix} a - 2a/n \\ 2a/n \end{bmatrix}$, where a two dimensional vector γ has a trinomial distribution with parameters $l, 1/n, 1/2 - 1/n$ and \tilde{X}^s denotes of a random variable X . So we can find that the distribution will have the positive mass around all points of the form $\pm ka$, $k = 0, 1, 2, \dots$.

As the last method considered here we will consider the normal approximation $N(0, \hat{\sigma}^2 \|v\|^2)$ of our original distribution, where

$$\hat{\sigma}^2 = s_{Y^\perp}^2 = \frac{(a - 2a/n)^2 + (l-1)(2a/n)^2}{n-1}.$$

We have then $\hat{\sigma}^2 \|v\|^2 \approx a^2(1 - 2/n)$ and this method fails completely in recovering the distribution of $v \cdot \epsilon | \mathcal{F}$, and of course the unconditional distribution may be far from normal.

7. Conclusion

We propose resampling from the permutations of residuals as a method of estimating the conditional sampling distributions of contrasts; without moment assumptions and assuming only exchangeable errors. The idea of resampling permutations of residuals occurs in [2], [7] and possibly elsewhere. Both papers place considerable emphasis on establishing conditions under which F -distributions associated with standard tests in regression are recovered by resampling permutations. In fact [2] does not otherwise recommend the method while [7] recommends it, but only as a *descriptive* method. Neither paper makes the crucial observation that the relations of lemmas 1, 2 imply that with probability tending to one $v \cdot \pi \epsilon^\perp | Y \approx v \cdot \epsilon | \mathcal{F}$ as $(d-1)/(n-1) \rightarrow 0$. In fact we have independently discovered lemmas 1, 2 in the course of extending the results of [8] which exploit resampling the signs of residuals in much the same way: $v \cdot \delta \epsilon^\perp | Y \approx v \cdot \epsilon | \mathcal{F}$ for symmetric errors. We believe that methods which approximately recover conditional distributions by resampling will be applicable to time series.

References

- [1] Bickel, P.J. and Freedman, D.A. (1984). Asymptotic normality and the bootstrap in stratified sampling. *Ann. Statist.* 12, 470-482.
- [2] Box, C.E.P. and Watson, G.S. (1962). Robustness to non-normality of regression tests. *Biometrika*, 49, 93-106.
- [3] Chambers, J.M., Cleveland, W.S., Kleiner, B. and Tukey, P.A. (1983). *Graphical methods for data analysis*. Belmont:Wadsworth.
- [4] Chernoff, H. (1973), Using faces to represent points in k -dimensional space. *J. Am. Statist. Assoc.*, 68, 361-368.
- [5] Efron, B. (1979), Bootstrap methods: Another look at the jackknife. *Ann. Statist.*, 7, 1-26.
- [6] Freedman, D.A. (1981). Bootstrapping regression models. *Ann. Statist.*, 9, 1218-1228.
- [7] Freedman, D.A. and Lane (1983). A nonstochastic interpretation of reported significance levels. *J. Business Econ. Statist.* 1, 292-298.
- [8] LePage, R. (1990), Bootstrapping signs. in *Exploring the limits of bootstrap*, R. LePage and L. Billard Eds., John Wiley and Sons Publ., (1992).
- [9] Wu, C.F.J. (1986), Jackknife, bootstrap and other resampling methods in regression analysis. *Ann. Statist.*, 14, 1261-1295.

Exact Logistic Regression: Theory, Applications and Software

Cyrus R. Mehta Nitin R. Patel

Department of Biostatistics, Harvard School of Public Health
and

Cytel Software Corporation
Cambridge, MA 02139

Abstract

We provide an alternative to the maximum likelihood method for making inferences about the parameters of the logistic regression model. The method is based on appropriate permutational distributions of sufficient statistics. It is useful for analysing small or imbalanced binary data with covariates. It is also applicable to small-sample clustered binary data. We illustrate the method by analysing several biomedical data sets with LogXact (1992), a new software package developed by Cytel Software Corporation.

1. Introduction

This paper deals with exact conditional inference for the parameters of the logistic regression model. It is customary to maximize the unconditional likelihood function for parameter estimation, and to perform hypothesis tests with either the Wald, the likelihood ratio or the efficient scores statistics. However for small or imbalanced data sets, and for highly stratified data, these asymptotic methods are unreliable. An alternative approach is to base the inference on exact permutational distributions of the sufficient statistics corresponding to the regression parameters of interest, conditional on fixing the sufficient statistics of the remaining parameters at their observed values. This approach was suggested by Cox (1971) but was not considered to be computationally feasible until fast algorithms for deriving these distributions were developed by Tritchler (1984), and Hirji, Mehta and Patel (1987), (1988), and Hirji (1992). A related asymptotic conditional approach was developed by Breslow and Day (1980) for logistic regression on matched sets. These investigators proposed treating each matched set as a separate stratum and eliminating all stratum-specific parameters from the likelihood function by conditioning on their sufficient statistics. The inference is then based on maximizing a conditional likelihood function. Although easier, computationally, than the exact permutational approach, conditional maximum likelihood estimation is not a trivial problem. Gail, Labin and Rubinstein (1983) developed a recursive algorithm to do the computations

efficiently. Their recursions have recently been generalized by Jajoo and Patel (1992) to encompass grouped data.

This paper describes the underlying theory for exact conditional inference, summarizes recent algorithmic developments that make this type of inference computationally feasible, and provides several illustrative examples that contrast exact conditional inference with the more customary unconditional maximum likelihood approach.

2. Models, Likelihood, and Sufficient Statistics

We consider two classes of models; logistic regression for unstratified binary data, and logistic regression for stratified binary data. In this section we discuss a uniform method of exact inference for both models, based on permutational distributions of appropriate sufficient statistics.

2.1. Logistic Regression for Unstratified Binary Data

Consider a set of independent binary random variables, Y_1, Y_2, \dots, Y_n . Corresponding to each random variable, Y_j , there is a $(p \times 1)$ vector $\mathbf{x}_j = (x_{1j}, x_{2j}, \dots, x_{pj})'$ of explanatory variables (or covariates). Let π_j be the probability that $Y_j = 1$. Logistic regression models the dependency of π_j on \mathbf{x}_j through the relationship

$$\log \left(\frac{\pi_j}{1 - \pi_j} \right) = \gamma + \mathbf{x}_j' \boldsymbol{\beta}, \quad (2.1)$$

where γ and $\boldsymbol{\beta} \equiv (\beta_1, \beta_2, \dots, \beta_p)'$ are unknown parameters. The likelihood function, or probability of an observed set of values, y_1, y_2, \dots, y_n , is

$$\Pr(Y_1 = y_1, Y_2 = y_2, \dots, Y_n = y_n) = \frac{\exp \left[\sum_{j=1}^n y_j (\mathbf{x}_j' \boldsymbol{\beta} + \gamma) \right]}{\prod_{j=1}^n [1 + \exp(\mathbf{x}_j' \boldsymbol{\beta} + \gamma)]} \quad (2.2)$$

The usual way to make inferences about $\boldsymbol{\beta}$ and γ is to maximize (2.2) with respect to these regression coefficients.

Suppose we are interested in inferences about β , and regard γ as a nuisance parameter. Then, instead of estimating γ from the above unconditional likelihood function, we can eliminate it by conditioning on the observed value of its sufficient statistic

$$m = \sum_{j=1}^n y_j.$$

This yields the conditional likelihood function

$$\Pr(Y_1 = y_1, Y_2 = y_2, \dots, Y_n = y_n | m) = \frac{\exp(\sum_{j=1}^n y_j x'_j \beta)}{\sum_R (\exp \sum_{j=1}^n y_j x'_j \beta)}, \quad \log \left(\frac{\pi_{ij}}{1 - \pi_{ij}} \right) = \gamma_i + x'_{ij} \beta, \quad (2.6)$$

where the outer summation in the denominator of (2.3) is over the set

$$R = \{(y_1, y_2, \dots, y_n) : \sum_{j=1}^n y_j = m\}.$$

Inference about β can now be approached in two ways, asymptotic and exact. An asymptotic approach is to maximize the conditional likelihood function (2.3). This would be a special case of the Breslow and Day (1980) method discussed in the next section for handling stratified data. Exact inference about β is based on the permutational distribution of its sufficient statistics. One can observe from the form of (2.3) that the $(p \times 1)$ vector of sufficient statistics for β is

$$t = \sum_{j=1}^n y_j x_j, \quad (2.4)$$

and its distribution is

$$\Pr(T_1 = t_1, T_2 = t_2, \dots, T_p = t_p) = \frac{c(t) e^{t' \beta}}{\sum_u c(u) e^{u' \beta}}, \quad (2.5)$$

where

$$c(t) = |S(t)|,$$

$$S(t) = \{(y_1, y_2, \dots, y_n) : \sum_{j=1}^n y_j = m, \sum_{j=1}^n y_j x_{ij} = t_i, i = 1, 2, \dots, p\},$$

$|S|$ denotes the number of distinct elements in the set S , and the summation in the denominator is over all u for which $c(u) \geq 1$. In other words, $c(t)$ is the count of the number of binary sequences of the form (y_1, y_2, \dots, y_n) which are such that $\sum_j y_j = m$ and $\sum_j y_j x_{ij} = t_i$ for $i = 1, 2, \dots, p$. Exact inference about β requires us to compute coefficients like $c(t)$ in which some of the sufficient statistics are fixed at their observed values and others are required to vary over their permissible ranges.

2.2. Logistic Regression for Stratified Binary Data

Suppose there are N strata, with binary responses in each of them. Let the i th stratum have m_i responses and $n_i - m_i$ non-responses. For all $1 \leq i \leq N$, and $1 \leq j \leq n_i$, let $Y_{ij} = 1$ if the j th individual in i th stratum responded; 0 otherwise. Define $\pi_{ij} = \Pr(Y_{ij} = 1 | x_{ij})$ where x_{ij} is a p -dimensional vector of covariates for the j th individual in the i th stratum. The logistic regression model for π_{ij} is of the form

where γ_i is a stratum specific scalar parameter and β is a $(p \times 1)$ vector of parameters common across all N strata. We are usually interested in inferences about β , and regard the γ_i 's as nuisance parameters. One could of course estimate these the nuisance parameters by the maximum likelihood method. However, Breslow and Day (1980) have shown that, for large N , this leads to inconsistent estimates. Moreover if there are too many regression coefficients, and the data set is small or imbalanced, the observed data frequently fall on the boundary of the likelihood function. When this happens the maximum likelihood estimates are undefined. (See, for example, Hirji, Tsiatis, Mehta, 1989). An alternative approach, proposed by Breslow and Day (1980), is to eliminate the nuisance parameters by conditioning on their sufficient statistics, in this case the number of responses, m_i , in each stratum. The conditional likelihood, or conditional probability of observing $Y_{ij} = y_{ij}, j = 1, 2, \dots, n_i, i = 1, 2, \dots, N$ is then

$$\frac{\exp \left[\sum_{i=1}^N \sum_{j=1}^{n_i} y_{ij} (x'_{ij} \beta) \right]}{\prod_{i=1}^N \sum_R \exp \left[\sum_{j=1}^{n_i} y_{ij} (x'_{ij} \beta) \right]}, \quad (2.7)$$

where the outer summation in the denominator is over the set

$$R_i = \{(Y_{i1}, \dots, Y_{in_i}) : \sum_{j=1}^{n_i} Y_{ij} = m_i\}.$$

Notice that the nuisance parameters, γ_i , have been factored out of the above conditional likelihood. The Breslow and Day (1980) approach is to make asymptotic inferences about β by maximizing (2.7). Exact inference is based on the sufficient statistics for β .

From (2.7) it is seen that the vector of sufficient statistics for β is

$$t = \sum_{i=1}^N \sum_{j=1}^{n_i} y_{ij} x_{ij}, \quad (2.8)$$

and its conditional distribution is

$$\Pr(T_1 = t_1, T_2 = t_2, \dots, T_p = t_p) = \frac{c(t)e^{\beta' t}}{\sum_u c(u)e^{\beta' u}},$$

where

$$c(t) = |Q(t)|,$$

$$Q(t) = \{(y_{ij}, j = 1, \dots, n_i, i = 1, \dots, N) : \sum_{i=1}^N \sum_{j=1}^{n_i} y_{ij} x_{ij} = t, \sum_{j=1}^{n_i} y_{ij} = m_i\},$$

$|Q|$ denotes the number of distinct elements in the set Q , and the summation in the denominator is over all u for which $c(u) \geq 1$. In other words $c(t)$ is the count of the number of ways of selecting the binary sequence $\{y_{ij}, i = 1, \dots, N, j = 1, \dots, n_i\}$ so as to satisfy the two conditions

$$\sum_{i=1}^N \sum_{j=1}^{n_i} y_{ij} x_{ij} = t,$$

and

$$\sum_{j=1}^{n_i} y_{ij} = m_i. \quad (2.9)$$

Notice that the distribution of T is of the same form for both stratified and unstratified logistic regression. This makes it possible to develop a single numerical algorithm for both cases.

3. Exact Conditional Inference

3.1. Conditional Inference for a Single Parameter

Suppose without loss of generality that we wish to make inferences about the single parameter β_p . By the sufficiency principle the conditional distribution of T_p given t_1, t_2, \dots, t_{p-1} depends only on β_p . Let $f(t_p | \beta_p)$ denote the conditional probability $\Pr(T_p = t_p | T_1 = t_1, \dots, T_{p-1} = t_{p-1})$. Then

$$f(t_p | \beta_p) = \frac{c(t_1, t_2, \dots, t_p)e^{\beta_p t_p}}{\sum_u c(t_1, t_2, \dots, t_{p-1}, u)e^{\beta_p u}}, \quad (3.1)$$

where the summation in the denominator is over all values of u for which $c(t_1, t_2, \dots, t_{p-1}, u) \geq 1$. Since this probability does not involve the nuisance parameters $(\beta_1, \beta_2, \dots, \beta_{p-1})$, it may be used for inference about β_p .

Hypothesis Testing: Suppose we wish to test

$$H_0: \beta_p = 0$$

against the two-sided alternative

$$H_1: \beta_p \neq 0.$$

The exact p-value is obtained by summing (3.1) over some specified critical region E :

$$p = \sum_{v \in E} f(v | \beta_p = 0). \quad (3.2)$$

The critical region E can be specified in different ways, leading to different types of tests. Two popular tests are the "conditional probabilities" test, and the "conditional scores" test. In the conditional probabilities test the critical region, denoted by E_{cp} , comprises of all values of the test statistic yielding a conditional probability no larger than the conditional probability at the observed value of t_p :

$$E_{cp} = \{v: f(v | \beta_p = 0) \leq f(t_p | \beta_p = 0)\}.$$

In the conditional scores test the critical region, denoted by E_{cs} , comprises of all values of the test statistic whose conditional scores equal or exceed the conditional score at the observed value of the test statistic:

$$E_{cs} = \{v: (v - \mu_p)^2 \sigma_p^{-2} \geq (t_p - \mu_p)^2 \sigma_p^{-2}\},$$

where μ_p and σ_p^2 are the mean and variance of T_p , based on its conditional distribution as specified by (3.1) at $\beta_p = 0$. For both types of exact tests we need an algorithm which can give us all the coefficients, $c(t_1, t_2, \dots, t_{p-1}, v)$, with t_1, t_2, \dots, t_{p-1} fixed at their observed values, and v varying over the entire range of T_p . Once we obtain these coefficients, computing the exact p-value is simply a matter of appropriate sorting and summing.

An asymptotic version of the conditional scores test is also possible. Here we obtain the p-value by referring the observed score, $(t_p - \mu_p) \sigma_p^{-1/2}$, to a chi-squared distribution on one degree of freedom. Note though that even for this asymptotic test it is necessary to compute the conditional mean, μ_p , and the conditional variance, σ_p . Asymptotic approximations to these conditional moments are available in Zelen (1991). We are currently developing a fast algorithm for obtaining the exact conditional moments.

Estimation: To obtain a level- α confidence interval, (β_-, β_+) for β_p , we invert the above test. Define

$$F_1(t_p | \beta) = \sum_{v \geq t_p} f(v | \beta)$$

and

$$F_2(t_p | \beta) = \sum_{v \leq t_p} f(v | \beta).$$

Let t_{\min} and t_{\max} be the smallest and largest possible values of t_p in the distribution (3.1). The lower confidence bound, β_- , is such that

$$F_1(t_p | \beta_-) = \alpha/2 \text{ if } t_{\min} < t_p \leq t_{\max},$$

$$\beta_- = -\infty \text{ if } t_p = t_{\min}.$$

Similarly the upper confidence bound, β_+ , is such that

$$F_2(t_p | \beta_+) = \alpha/2 \text{ if } t_{\min} \leq t_p < t_{\max},$$

$$\beta_+ = \infty \text{ if } t_p = t_{\max}.$$

One can show that this definition does indeed produce an interval, and the interval is guaranteed to have the desired $(100)(1 - \alpha)\%$ coverage for β_p .

A point estimate for β_p may be computed in two ways. The conditional maximum likelihood estimate, β_{cmle} , is obtained by maximizing $f(t_p | \beta)$ by choice of β . However if either $t_p = t_{\min}$, or if $t_p = t_{\max}$, β_{cmle} is undefined, as the likelihood function cannot be maximized. An alternative estimate for β_p that has several useful properties (see, for example, Hirji, Tsiatis, Mehta, *American Statistician*, vol 43, 1, 1988) is the median unbiased estimate

$$\beta_{mue} = (\beta_+ + \beta_-)/2,$$

where β_- and β_+ are evaluated at a confidence level $\alpha = 0.5$. If $\beta_- = -\infty$, we define $\beta_{mue} = \beta_+$, while if $\beta_+ = \infty$, we define $\beta_{mue} = \beta_-$. Thus, unlike the maximum likelihood estimate, the median unbiased estimate is always defined, even at the extreme points of the sample space.

3.2. Conditional Inference for Several Parameters

To make inferences about several parameters simultaneously we need the joint distribution of their sufficient statistics conditional on the observed values of the remaining sufficient statistics. Suppose we partition the $(p \times 1)$ vector of regression parameters β into two parts; a $(p_1 \times 1)$ component, β_1 , and a $(p_2 \times 1)$ component, β_2 . Let \mathbf{t}_1 and \mathbf{t}_2 be the corresponding vectors of sufficient statistics. We wish to test the null hypothesis

$$H_0: \beta_2 = 0$$

against the two-sided alternative that at least one of the elements of β_2 is not 0. By the sufficiency principle the conditional distribution of \mathbf{T}_2 given $\mathbf{T}_1 = \mathbf{t}_1$ is free of the nuisance parameters β_1 . Thus we denote the conditional probability $\Pr(\mathbf{T}_2 = \mathbf{t}_2 | \mathbf{T}_1 = \mathbf{t}_1)$ by $f(\mathbf{t}_2 | \beta_2)$, where

$$f(\mathbf{t}_2 | \beta_2) = \frac{c(\mathbf{t}_1, \mathbf{t}_2)e^{\beta_2' \mathbf{t}_2}}{\sum_{\mathbf{u}} c(\mathbf{t}_1, \mathbf{u})e^{\beta_2' \mathbf{u}}}. \quad (3.3)$$

The summation in the denominator of (3.3) is taken over all values of \mathbf{u} for which $c(\mathbf{t}_1, \mathbf{u}) \geq 1$.

The exact two-sided p-value for testing H_0 is obtained by summing (3.3) over some critical region E :

$$p = \sum_{\mathbf{v} \in E} f(\mathbf{v} | \beta_2 = 0). \quad (3.4)$$

Again we have two types of critical regions leading respectively to the conditional probabilities test and the conditional scores test. The critical region for the conditional probabilities test is

$$E_{cp} = \{\mathbf{v}: f(\mathbf{v} | \beta_2 = 0) \leq f(\mathbf{t}_2 | \beta_2 = 0)\}.$$

The critical region for the conditional scores test is

$$E_{cs} = \{\mathbf{v}: (\mathbf{v} - \mu_2)' \Sigma_2^{-1} (\mathbf{v} - \mu_2) \geq (\mathbf{t}_2 - \mu_2)' \Sigma_2^{-1} (\mathbf{t}_2 - \mu_2)\}.$$

μ_2 is the mean, and Σ_2 is the variance covariance matrix of $f(\mathbf{t}_2 | \beta_2 = 0)$. For both types of tests we need an algorithm that can give us all the coefficients $c(\mathbf{t}_1, \mathbf{v})$ with \mathbf{t}_1 fixed and \mathbf{v} varying over the entire range of \mathbf{T}_2 .

An asymptotic version of the conditional scores test is obtained by referring the scores statistic $(\mathbf{t}_2 - \mu_2)' \Sigma_2^{-1} (\mathbf{t}_2 - \mu_2)$ to a chi-squared distribution on $(p_2 - 1)$ degrees of freedom.

4. Numerical Algorithms

We will confine ourselves to referencing the most recent algorithmic developments for exact logistic regression, rather than describing these algorithms in detail here. Bayer and Cox (1979) developed an early algorithm in which all possible binary sequences of the Y variable are enumerated exhaustively. Tritchler (1984) provided a substantial improvement relative to exhaustive enumeration, using a specific application of the inverse Fourier transform algorithm of Pagano and Tritchler (1983). However Tritchler's algorithm is only applicable to models with a single covariate, with possible stratification for matched sets. Hirji, Mehta and Patel (1987) developed a general and efficient algorithm for the evaluating the permutational distribution of $\mathbf{T}_2 | \mathbf{T}_1 = \mathbf{t}_1$ (see equation (3.3)) for unstratified data, and subsequently extended it to the stratified case (1988). Their algorithms are incorporated into the LogXact, an new statistical package for exact logistic regression. We have used LogXact to generate the examples in the next section. Hirji (1992) has recently extended these algorithms further to allow for polytomous regression.

5. Examples

The examples in this section were all provided by investigators who were either unable to perform the conventional logistic regression analysis on their data, or who distrusted the results of the conventional analysis, because of the small sample sizes.

5.1. Advance Indicators of HIV Infection in Infants

We are grateful to Dr Shengan Lai, University of Miami, for providing this example. A hospital based prospective study of perinatal infection and human immunodeficiency virus (HIV-1) by Hutto, Parks, Lai, et. al. (1991) investigated, among other things, the possibility that the CD4 and CD8 blood serum levels measured in infants at 6 months of age might be predictive for their eventually developing a HIV infection. The data on HIV infection rates and blood serum levels are tabulated below.

We wish to determine if the CD4 and CD8 serum levels are statistically significant in the logistic regression model $HIV = CD4 + CD8$. Now although each of these covariate is at three ordered levels (0, 1, 2), it was felt that they should be included in the regression model as qualitative or "factor" variables, rather than as quantitative variables. Otherwise one would have to assume, erroneously, that 0, 1, and 2 were the actual numerically observed blood counts. This requires CD4 and CD8 to each be split up into two dummy variables (0 versus 2, and 1 versus 2) in the regression model. The model may be specified formally as:

$$\log \left(\frac{\pi_j}{1 - \pi_j} \right) = \gamma + \sum_{l=1}^4 \beta_l x_{lj}, \quad (5.1)$$

where, for the j th subject, $x_{1j} = 1$ if CD4 is at level 0 and 0 otherwise; $x_{2j} = 1$ if CD4 is at level 1 and 0 otherwise; $x_{3j} = 1$ if CD8 is at level 0 and 0 otherwise; $x_{4j} = 1$ if CD8 is at level 1 and 0 otherwise.

As is often the case with small and imbalanced data sets, the regression parameters in model (5.1) cannot be estimated by the maximum likelihood method because the observed data fall on the boundary of the sample space; you will find that conventional packages like BMDP, GLIM, EGRET, SAS, or SYSTAT, are unable to produce any output. Nevertheless the observed rates of HIV infection do vary considerably with the serum levels and formal tests of significance would be useful. The exact conditional distributions of appropriate sufficient statistics enable us to perform such tests. To determine if the CD8 levels are predictive of HIV infection we must test the null hypothesis

$$H_0: \beta_3 = \beta_4 = 0.$$

Since the maximum likelihood method fails to converge, the usual 2 degree-of-freedom likelihood ratio, Wald, and scores tests are undefined. However the joint and conditional distributions of the sufficient statistics are well defined and can be used for the exact inference. The

sufficient statistic for β_l is $T_l = \sum x_{lj} Y_j$, and the sufficient statistic for the constant term is $T_0 = \sum Y_j$, the summation being taken over all subjects. An exact test of H_0 is based on $f(t_3, t_4 | \beta_3 = \beta_4 = 0)$, the null permutational distribution of (T_3, T_4) given that the remaining sufficient statistics are fixed at their observed values; i.e., $(T_0 = 14, T_1 = 5, T_2 = 8)$. This distribution was computed by LogXact.

For testing H_0 , we may use either the conditional probability test or the conditional scores test. The observed value of (t_3, t_4) is (6, 4). Its corresponding conditional probability is $f(6, 4 | \beta_3 = \beta_4 = 0) = 0.00532$. Thus the critical region for the conditional probability test, E_{cp} , consists of all (t_3, t_4) points in the sample space with probabilities less than or equal to 0.00532. The exact p-value is

$$p_{cp} = \sum_{E_{cp}} f(t_3, t_4 | \beta_3 = \beta_4 = 0) = 0.0323$$

An alternative exact test for H_0 is the conditional scores test. For each (t_3, t_4) in the sample space of the conditional distribution, one can compute a conditional score of the form

$$q = ((t_3, t_4) - (\mu_3, \mu_4)) \Sigma_{3,4}^{-1} ((t_3, t_4) - (\mu_3, \mu_4))'$$

where μ_3 is the mean of T_3 , μ_4 is the mean of T_4 and $\Sigma_{3,4}$ is the variance-covariance matrix of $f(t_3, t_4 | \beta_3 = \beta_4 = 0)$. The observed conditional score is $q = 7.293$. Thus the critical region for the conditional scores test, E_{cs} , consists of all (t_3, t_4) points in the sample space with conditional scores greater than or equal to 0.7.293. The exact p-value is

$$p_{cs} = \sum_{E_{cs}} f(t_3, t_4 | \beta_3 = \beta_4 = 0) = 0.0256$$

Below we have tabulated the exact p-values for CD4 and CD8, based on both the conditional probability and conditional scores tests. Asymptotic p-values based on 2-degree-of-freedom chi-squared analogs of the conditional scores tests are also reported.

The two exact tests give similar answers and both demonstrate the important role of the CD4 and CD8 counts in predicting HIV infection. In the above table we have also reported an asymptotic p-value for the conditional scores test. This p-value is the area to the right of the observed conditional score from a chi-squared distribution with 2 degrees of freedom. For example, the observed conditional score for testing CD8 was 7.293. Thus the asymptotic p-value is the area to the right of 7.293 from a chi-squared distribution with 2 degrees of freedom; i.e. 0.0261. We notice that the asymptotic

Proportion Developing HIV	Serum Levels at 6 Months	
	CD4	CD8
1/1 (100%)	0	2
2/2 (100%)	1	2
4/7 (57%)	0	0
4/12 (33%)	1	1
1/3 (33%)	2	2
2/7 (29%)	1	0
0/2 (0%)	2	0
0/13 (0%)	2	1

Type of Test	Exact P-Values		Asymptotic P-Values	
	CD4	CD8	CD4	CD8
Conditional Probability Test	0.009	0.0323	—	—
Conditional Scores Test	0.007	0.0256	0.0095	0.0261

conditional scores tests are very accurate. They closely match the corresponding exact tests. However in order to compute the conditional scores we actually needed the exact conditional moments of $f(t_3, t_4 | \beta_3 = \beta_4 = 0)$. These moments could only be derived from the exact conditional distribution. Thus there is at present no computational advantage in substituting these asymptotic p-values for the exact ones. A fruitful area of research would be to obtain accurate asymptotic moments for the conditional distributions of the sufficient statistics of logistic regression parameters.

5.2. Schizophrenia and Birth Complications

We thank Dr. Armando Garsd for providing this example. A case-control study (Garsd et. al., 1988) was designed to determine the role of birth complications in schizophrenics. The sample consisted of 7 families with several siblings per family. An individual within a family was classified either as normal or schizophrenic. A "birth-complications index" was available for each individual, ranging in value from 0 (uncomplicated birth) to 15 (severely complicated birth). The data are displayed below:

Is there a positive correlation between the chance of schizophrenia and the birth-complications index? The data do indeed suggest some such tendency. But the numbers are small, and the magnitude of the effect appears to vary across families. This is an ideal situation for exact logistic regression on matched sets. Treating each family as a separate matched set, one can model π_{ij} , the probability of schizophrenia for the j th sibling in the

i th family in terms of the birth-complications index, x_{ij} :

$$\log \left(\frac{\pi_{ij}}{1 - \pi_{ij}} \right) = \gamma_i + \beta x_{ij}.$$

We eliminate nuisance parameter γ_i , corresponding to the family effect, by conditioning on the total number of schizophrenics within each family. We then estimate β by the methods of Section 2.2. The results are tabulated below.

For this small data-set there are noticeable p-value differences between the exact conditional scores test and the Wald or Likelihood Ratio asymptotic tests. On the other hand, the p-values for the exact and asymptotic conditional scores tests are very similar. This is what we observed in the HIV example also. While this suggests that one could get away with using the asymptotic conditional scores test rather than its exact counterpart, there is, as explained previously, actually no computational advantage to doing so. In order to compute the conditional scores statistic one needs the mean and variance of the conditional distribution of the sufficient statistic for β . So far no accurate method for estimating these conditional moments is available, short of actually generating the entire permutational distribution of the sufficient statistic. In that case it is just as easy to perform the exact test.

5.3. Cross-over Clinical Trial of Analgesic Efficacy

The data below are taken from a three-treatment, three-period cross-over clinical trial. The three drugs are A=New Drug, B=Aspirin, C=Placebo. The primary end-point was analgesic efficacy, here dichotomized as

Family ID	Birth-Complications Index	Number of Siblings		
		Normal	Schizophrenic	Total
1	15	0	1	1
1	7	1	0	1
1	6	1	0	1
1	5	1	0	1
1	3	2	0	2
1	2	3	0	3
1	0	1	0	1
2	2	0	1	1
2	0	1	0	1
3	9	0	1	1
3	2	1	0	1
3	1	1	0	1
4	2	0	1	1
4	0	4	0	4
5	6	1	0	1
5	3	0	1	1
5	0	0	1	1
6	3	1	0	1
6	0	3	1	4
7	6	0	1	1
7	2	1	0	1

Inference for Beta	
Conditional Maximum Likelihood Estimate	0.325
Exact 95% Confidence Interval	(0.0223 to 0.741)
Asymptotic 95% Confidence Interval	(-0.004 to 0.654)
Exact P-Value (Conditional Scores)	0.0167
Asymptotic P-Value (Conditional Scores)	0.0129
Asymptotic P-value (Wald)	0.0528
Asymptotic P-value (Likelihood Ratio)	0.023

0 for relief and 1 for no-relief. See Snapinn and Small (*Biometrics*, 42, 583-592, 1986) for details.

The question to be addressed is whether the three treatments are different. We answer this question by including treatment as the primary covariate in a logistic regression model for matched sets. In this model, treatment is included as an unordered categorical covariate at three levels, and hence, with two degrees of freedom. We regard each patient as a matched set. Within such a matched set there are three observed responses, one at each of the three time periods P1, P2 and P3. Now although these responses are all on the same patient, and are therefore dependent, we assume that this dependence can be removed by appropriate modelling. See Jones and Kenward (*Statistics in Medicine*, 6, 555-564, 1987). For the present data set we will assume that the three

response probabilities within a matched set may be regarded as independent if they arise in a logistic regression model containing a covariate term for the effect of cross-over. The cross-over term will have four levels (induction, cross-over from treatment A, cross-over from treatment B, cross-over from treatment C), and three degrees of freedom. Technically the model should also include a two degree of freedom covariate term for the period effect. However, for this small data set, the period effect and the cross-over effect are aliased. The model may thus be specified as:

$$\log \frac{\pi_{jkl}}{1 - \pi_{jkl}} = \gamma_j + \beta_k + \gamma_l,$$

where γ_j is the stratum effect for the j th matched set (or subject), β_k is the effect of drug k , and γ_l is the effect of the l th cross-over level.

Patient	Drug	Response		
		P1	P2	P3
1	ABC	0	1	1
7	ABC	0	1	1
2	BCA	0	1	1
8	BCA	0	0	0
3	CAB	1	0	0
9	CAB	1	0	1
4	CBA	1	0	1
10	CBA	1	0	0
5	ACB	0	0	0
11	ACB	0	1	0
6	BAC	1	0	0
12	BAC	0	0	1

The following results are obtained for the two degree of freedom test that there is no treatment effect:

Thus it appears as though the conditional scores test fits the exact test very well, but there are wide variations among the other asymptotic tests.

5.4. Predictors of Disease Free Survival for Osteogenic Sarcoma

In a 46-patient study of non-metastatic osteogenic sarcoma, (Goorin et. al., *J. Clin. Oncology*, 5, 1987), the investigators were interested in determining the predictors for a three year disease free interval (DFI3). The covariates of interest were gender (SEX), any osteoid pathology (AOP), and lymphocytic infiltration (LI). The data are displayed below, one covariate at a time, and overall.

Osteogenic Sarcoma Data: Covariate by Covariate

DFI3	Lymphocytic Infiltration?		Total
	No	Yes	
No	0	17	17
Yes	10	19	29
Total	10	36	46

DFI3	Sex		Total
	Female	Male	
No	2	15	17
Yes	13	16	29
Total	15	31	46

Marginally, the effect of LI on DFI3 is certainly statistically significant; no subject with lymphocytic infiltration had a three year disease free interval. Unfortunately if this covariate is included in the logistic regres-

DFI3	Any Osteoid Pathology?		Total
	No	Yes	
No	4	13	17
Yes	17	12	29
Total	21	25	46

sion model, along with SEX and AOP, the maximum likelihood estimates of the regression parameters do not exist. All the conventional logistic regression packages fail on this data set. Exact inference is possible however, and does provide new insight about the data.

Let t_1, t_2, t_3, t_4 be the sufficient statistics corresponding to LI, SEX, AOP, and the constant term, respectively. Note that t_i is just the sum of covariate- i values over all subjects with a three year disease free interval. Thus $t_1 = 19$, $t_2 = 16$, $t_3 = 12$, and $t_4 = 29$. The distribution of counts.

$$c(t_1, t_2 = 16, t_3 = 12, t_4 = 29)$$

for all possible values of t_1 is displayed below.

t_1	$c(t_1, 16, 12, 29)$
19	29,445,360
20	147,312,480
21	271,271,448
22	231,819,344
23	95,325,644
24	17,473,144
25	1,204,008
26	19,448
Total	793,870,896

It is interesting to notice

Type of Test	Chi Squared Value	P-value
Likelihood Ratio	5.6	.06
Bivariate Wald	2.47	.291
Unconditional Scores	4.56	.099
Conditional Scores	3.684	.1585
Exact	3.684	.1525

DFI3	LI	SEX	AOP	Frequency Count
1	0	0	0	3
1	0	0	1	2
1	0	1	0	4
1	0	1	1	1
1	1	0	0	5
1	1	0	1	3
1	1	1	0	5
1	1	1	1	6
0	1	0	1	2
0	1	1	0	4
0	1	1	1	11

Table 1: Osteogenic Sarcoma Data; Overall

- There are nearly 800 million binary sequences of the y_j values which are such $t_2 = 16$, $t_3 = 12$, and $t_4 = 29$. And yet they all "club" into barely eight distinct values of t_1 . A good algorithm must exploit this clubbing, because otherwise, exhaustive enumeration of all possible binary sequences would be computationally explosive.
- The exact conditional distribution of T_1 is extremely asymmetric. Normal approximations would not work too well, though Edgeworth and saddlepoint approximations might be worth trying.
- The observed value, $t_1 = 19$, is at the minimum of its range. This is the reason for the failure of the maximum likelihood method to produce estimates of the regression parameters.

Exact inference about the parameter corresponding to LI is now straightforward. The exact one-sided p-value for testing $\beta_1 = 0$ is

$$p = 29445360/793870896 = 0.037$$

Since t_1 is at its minimum value, the lower 95% confidence bound for β_1 is $-\infty$. The upper 95% confidence bound, β_+ , is the solution to

$$\frac{c(19, 16, 12, 29)e^{19\beta_+}}{\sum_{t_1=19}^{26} c(t_1, 16, 12, 29)e^{t_1\beta_+}} = 0.025.$$

Binary search rapidly yields $\beta_+ = 0.161$.

The conditional distributions of the other sufficient statistics and the corresponding parameter estimates can be obtained similarly.

5.5. Bupenorphine Treatment for Drug Addicts

We thank Dr. Edward Lee, Substance Abuse Treatment Unit, Department of Psychiatry, Yale University, for providing this example of multiple binary responses on five substance abusers. The five individuals were treated with both Placebo ($X = 1$) and Bupenorphine ($X = 2$) at each of four doses (0, .125, .250, .500 mg/m²). The binary response measured at each dose level was presence/absence of abnormal heart-beat ($Y = 1/0$). The data are displayed below:

The question of interest was, whether Bupenorphine increased the probability of abnormal heart-beat relative to placebo. The data were complicated by the fact that each individual was treated several times, at different dose levels of both treatments, thereby providing a sequence of clustered binary responses. We handled this problem by using stratified logistic regression, regarding each individual as a separate stratum. The model was thus

$$\log \left(\frac{\pi_{ij}}{1 - \pi_{ij}} \right) = \gamma_i + \beta_1 x_{1ij} + \beta_2 x_{2ij},$$

where i indexes the strata, $i = 1, 2, \dots, 5$, j indexes the

Patient	Dose	Response	Treatment	Patient	Dose	Response	Treatment
1	0	0	1	1	0	1	2
2	0	0	1	2	0	0	2
3	0	0	1	3	0	1	2
4	0	0	1	4	0	0	2
5	0	0	1	5	0	0	2
1	125	0	1	1	125	1	2
2	125	1	1	2	125	1	2
3	125	0	1	3	125	1	2
4	125	0	1	4	125	1	2
5	125	1	1	5	125	1	2
1	250	1	1	1	250	1	2
2	250	1	1	2	250	1	2
3	250	1	1	3	250	1	2
4	250	1	1	4	250	1	2
5	250	1	1	5	250	1	2
1	500	1	1	1	500	1	2
2	500	1	1	2	500	1	2
3	500	1	1	3	500	1	2
4	500	1	1	4	500	1	2
5	500	1	1	5	500	1	2

different dose levels within each stratum, $j = 1, 2, \dots, 4$, x_{1ij} is the j th drug dose in the i th stratum, and x_{2ij} is the j th treatment in the i th stratum (1 for placebo; 2 for Bupenorphine).

As with several other examples presented in this paper, the maximum likelihood method failed to produce estimates of the regression coefficients. However the exact method, based on the permutation distribution of the sufficient statistics produced the following results:

The exact method reveals that Bupenorphine does indeed induce a statistically significant increase in abnormal heart beat, after adjusting for the effects of clustering, and varying dose levels.

6. Conclusions

We have provided a way to analyse small-sample binary data with covariates, and have illustrated our approach through several examples that could not be analysed by conventional methods of logistic regression. For data in the form of independent binary observations we use the unstratified logistic regression model, and base our inference on appropriate permutational distributions of the sufficient statistics. For clustered binary data, consisting of a few experimental units, and repeated binary observations on each unit, we use the stratified logistic regression model. Each unit is treated as a separate stratum or matched set. The inference on the regression parameters proceeds as before and is based on

permutational distributions of sufficient statistics. The permutational approach for clustered binary data is a useful complement to the generalized estimating equations approach (Zeger and Liang, 1986), for it is valid in small samples, while the latter is valid in large samples.

We have seen that with small and imbalanced data the maximum likelihood approach may fail, even though the covariates in the model are statistically significant. The permutational approach on the other hand provides valid inferences for this situation.

We have identified one useful problem for future research. The asymptotic conditional scores test yields p -values that are very close to corresponding exact p -values. If one could compute the conditional scores quickly, one could simply refer them to appropriate chi-squared distributions and thereby obtain very accurate p -values without the need to derive complicated permutational distributions of sufficient statistics. Some work along these lines is available in Zelen (1991).

Software for exact logistic regression is available in the LogXact (1992) software package, distributed by Cytel Software Corporation, Cambridge, MA.

7. References

- Breslow NE, Day NE (1980). *Stat Methods in Cancer Research*. IARC, Lyon.
- Byar L, Cox C (1979). Algorithm AS142. *App. Stat.*

Parameter	Point Estimate	Exact 95% CI	Exact P-Value
β_1	0.0210	0.00825 to infinity	5.11×10^{-6}
β_2	2.22	0.0977 to infinity	0.0396

28, 319-24.

Cox, DR (1970). *Analysis of Binary Data*. Chapman and Hall, London.

Gail MH, Lubin JH, and Rubenstein LV (1981). Likelihood calculations for matched case-control studies and survival studies with tied death times. *Biometrika*, 68, 703-707.

Garsd, A. (1988). Schizophrenia and birth complications. Unpublished manuscript.

Hirji KF, Mehta CR, Patel NR (1987). Computing distributions for exact logistic regression. *JASA*, 82, 1110-1117.

Hirji KF, Mehta CR, Patel NR (1988). Exact inference for matched case-control studies. *Biometrics*, 44, 803-814.

Hirji KF, Tsiatis AA, Mehta CR (1989). Median unbiased estimation for binary data. *The American Statistician*, 43, 7-11.

Hirji KF (1992). Exact distributions for polytomous data. *JASA*, in press.

Hutto C, Parks WP, Lai S (1991). A hospital based prospective study of perinatal infection with HIV-1. *J. Pdiatr.*, 118, 347-53.

Jones B, Kenward MG (1987). Binary data from a three-period trial. *Stats. in Med.*, 6, 555-64.

LogXact (1992). Cytel Software Corporation, Cambridge, MA.

Pagano M, Tritchler D (1983). Permutation distributions in polynomial time. *JASA*, 435-40.

Patel NR, Jajoo B (1992). Recursions for conditional likelihood computation. Manuscript.

Snapinn SM, Small RD (1986). Reg models for categorical data. *Biometrics*, 42, 583-92.

Tritchler D (1984). An algorithm for exact logistic regression. *JASA*, 79, 709-711.

Zeger SL, Liang KY (1986). Longitudinal analysis. *Biometrics*, 42, 121-130.

Zelen M (1991). Multinomial response models. *Comp.Stat. Data Analysis*, 12, 249-254.

Visualizing Correlation Decision Making in Data Fusion Problems

Donald E. Brown, John F. Elder, IV, and Clarence Louis Pittard, Jr.
 Institute for Parallel Computation &
 Department of Systems Engineering
 University of Virginia
 Charlottesville, Va. 22903-2442

Abstract

Modern surveillance and monitoring activities in manufacturing, medicine, meteorology, and the military frequently require processing data from multiple sensors. The incoming data must be associated or correlated with previously observed data to produce an estimate of the current environmental state. The correlation decision determines if the incoming data represent changes in the environment; if they do, the environmental state is updated to reflect the changes. Operational approaches to this problem generally employ linear discriminant functions. This paper shows the results of applying visual data analysis to the evaluation of competing approaches to data correlation. The specific application examined is the location of radars using data from superheterodyne receivers. A simulation of the performance of these receivers implemented on a multiprocessor is used to generate the data. Three dimensional projections in feature space show clearly the deficiencies of the linear discriminant approach. At the same time, the projections provide a clear indication of the minimal error obtainable through nonlinear methods. A video demonstration of these results is available.

1. Introduction

Data correlation lies at the heart of most modern surveillance and monitoring systems. Data from multiple sensors is continually correlated or merged with existing data to maintain an estimate of the changing environmental state.

Applications of this type of correlation problem exist in a number of fields. For example, in medicine, sensors monitor the current state of a patient and physicians attempt to use this information to detect changes resulting from either the application of treatment or the onset of disease. In the military and arms control, sensors monitor the current disposition of forces and analysts seek to detect changes that foreshadow specific military operations. In manufacturing, sensors and human operators monitor the condition of products and subsystems passing through an assembly process to detect changes in quality. The correlation decision is typically made by a human; however, as sensors have become less expensive they have become more numerous and easy to employ. Hence, most fields face an explosion in the amount of data that must be correlated, which has necessitated the deployment of automated correlation systems.

Current operational approaches to data correlation employ linear discriminant functions, although the elements in the discriminant function might be nonlinear in the incoming data. Problems with this approach, particularly in arms control monitoring and military surveillance, have led to interest in improved methods. This paper focuses on the data correlation problem in emitter identification, and describes our use of visual data analysis to suggest improvements to existing algorithms. The next section defines the data correlation problem and introduces the specific emitter identification application studied. Section 3 presents our testing results and data analysis, and Section 4 contains our conclusions.

2. Data Correlation

Let $E(t)$ be the environment of interest at time t (Figure 1). Θ is the sensor function that maps from $E(t)$ into a set of measurements $x(t) = (x_1(t), \dots, x_n(t))$. $R(t)$ is the representation of the environment in the machine at time t . $R(t)$ can include as many of the past sensor reports as machine address space and algorithm performance allow (see Barker, et al. 1991, for a complete discussion of these issues). The correlation algorithm, A , acts on $R(t)$ to produce an estimate of $E(t)$. Hence, the data correlation problem is to design A to optimally estimate $E(t)$. (Spillane, et al. 1989 discuss the notion of optimality for this problem.)

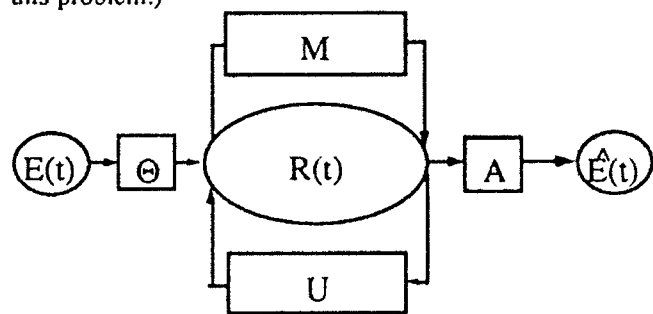


Figure 1: The Data Correlation Process

There is no known way to formally prove the superiority of one correlation algorithm over another across all application domains, so testing in the domain remains the basis for algorithm comparisons. Here, we examine data correlation for the problem of emitter identification.

For emitter identification we want to locate ground-based radars and transceivers; hence, $E(t)$ consists of point sources in a plane, and $x(t)$ contains location measurements, error estimates, and electronic parameters (e.g., pulse repetition interval). For our analysis we modeled the performance of airborne sensors (superheterodyne receivers) detecting four types of ground-based radars. Each of the four radars could operate in one of five modes (e.g., different frequency settings). Two hundred of these radars were randomly placed in a 120 x 80 km. region according to a uniform distribution.

Our simulation allowed us to judge the performance of correlation decisions, since (in this test problem) we know whether sensor reports should correlate or not (i.e. we know whether they were produced by the same or different radars). Competing algorithms were judged by their accuracy on classifying pairs of reports using 24 hours of simulated sensor operation (about 4000 sensor reports).

3. Data Correlation Tests and Visual Analysis

The current approach to data correlation of emitters uses a linear discriminant function (Wright, 1980), although each element of the weighted sum can be nonlinear in the sensor report measurements. Three electronic factors are available: pulse repetition interval (PRI), pulse duration (PD), and frequency. The fourth input, Euclidean distance information, is monotonically transformed by a chi-squared function to obtain a measure of similarity (0: close - 1: far).

As illustrated in Figure 2, this classifier is a "perceptron" form -- linearly weighted factors summed and submitted to a threshold. This approach has a number of problems (see Brown, et al. 1992). Our concern here was the sources of correlation decision errors.

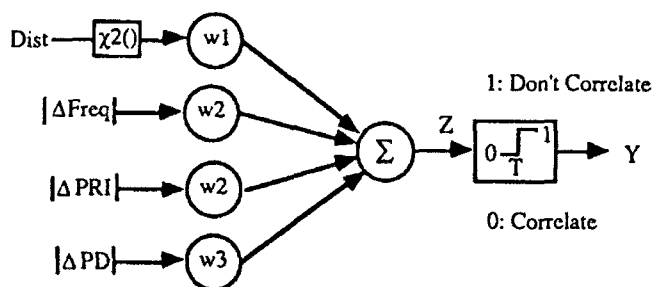


Figure 2: Linear Method takes Perceptron Form

Figures 3a and 3b are two-dimensional projections (of a 4-d space) of the pairs of sensor reports, highlighting the errors made by the linear discriminant approach ("I": correctly classified correlated pair; ".": correctly classified uncorrelated pair; "o": misclassified pair). The view of Figure 3a, with the three electronic parameters, is not inconsistent with the "diamond shape" implicitly assumed of

the classes by use of the linear (L_1) discriminate function. However, including the similarity metric, as in Figure 3b, is. In Figure 3a, missclassification errors appear on the "hull" of the center of the space, surrounding the correlated reports, as would be sought (confusion only on the boundaries). In Figure 3b, the errors are more obvious.

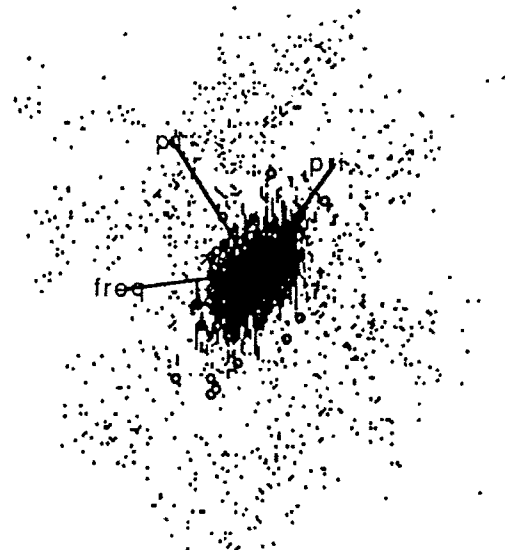


Figure 3a: Projection of Electronic Factors

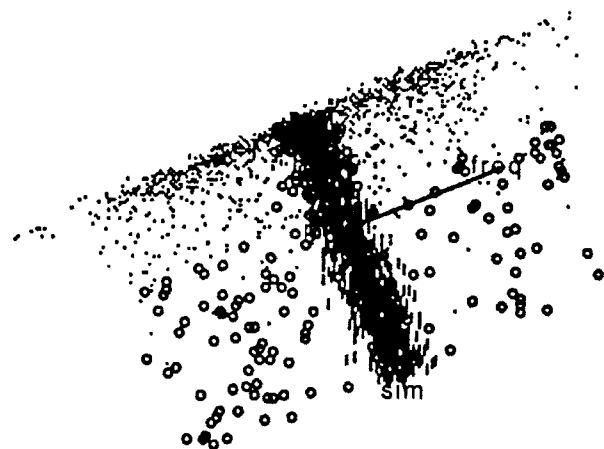


Figure 3b: Projection with Similarity Measure

Note that only a subset of cases are shown. Since all features are difference magnitudes (i.e., non-negative), roughly half the values were randomly assigned negative signs, to improve visualization. Correlated pairs, which normally occur only a fraction of the time, are here about half the cases (3401 out of 6912). This balance was made possible through use of a *pre-filter* capable of discarding *a priori* all pairs with extremely small correlation probabilities. Thereby, all clearly mismatched pairs can be removed from the data (deterministically, and with zero

error), preventing the classification stage from being swamped by cases of one type -- an important consideration for empirical methods.

To investigate the factor distributions further, Figures 4-6 show the marginal distributions for the pairs of reports that correlated (a), and those that did not (b), for PRI, PD, and the similarity measure.

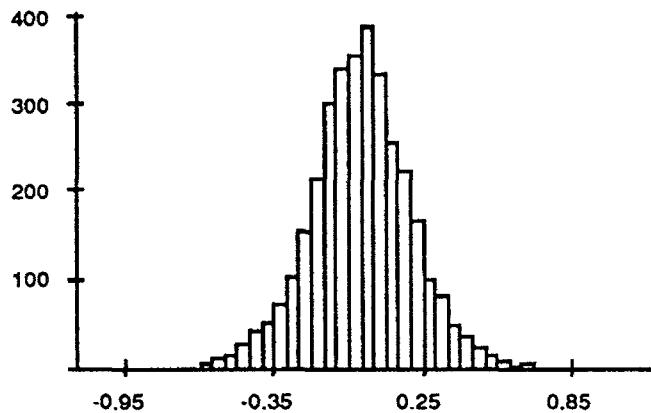


Figure 4a: Marginal Distribution for Correlated Reports for Pulse Repetition Interval

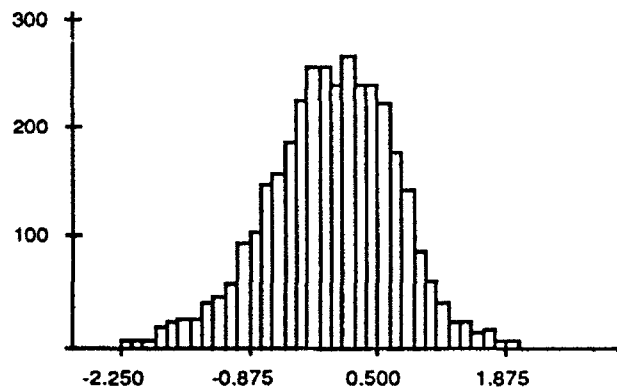


Figure 4b: Marginal Distribution for Uncorrelated Reports - Pulse Repetition Interval

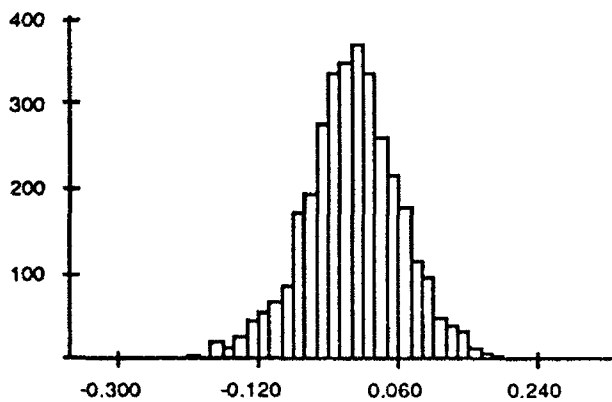


Figure 5a: Marginal Distribution for Correlated Reports for Pulse Duration

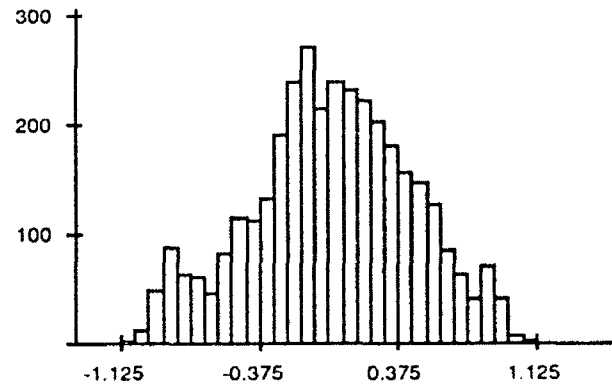


Figure 5b: Marginal Distribution for Uncorrelated Reports for Pulse Duration

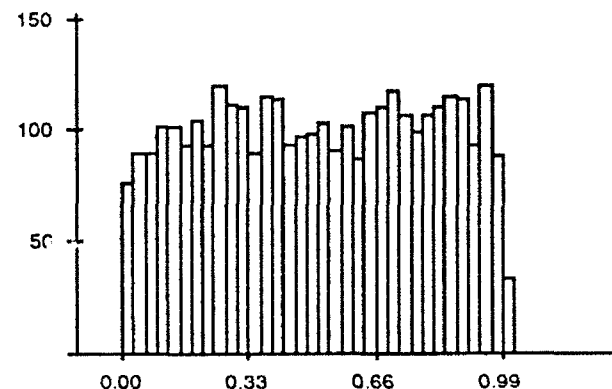


Figure 6a: Marginal Distribution for Correlated Reports for Similarity

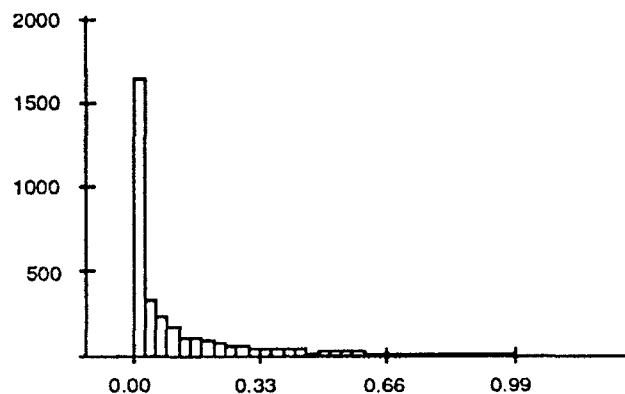


Figure 6b: Marginal Distribution for Uncorrelated Reports for Similarity

Note that the conditional electronic distributions (Figures 4 and 5) have roughly the same (Gaussian) shape, but a lower variance for the correlated cases (a) than for the uncorrelated cases (b). (Were all parameters like this, the conventional "figure of merit", or linear discriminant, technique of Figure 2 might prove competitive.) The distributions of the similarity metrics however, are strikingly different: That conditioned on correlated pairs (Figure 6a) is roughly uniform[0,1], while the uncorrelated

distribution (6b) is exponential in shape. Given this information, we investigated several alternatives to linear discriminants, and two in detail: 1) a parametric approach using the marginal distributions derived from the training set and 2) logistic regression. The results are shown in Table 1 (where 0 now refers to uncorrelated cases; 1 to correlated).

Table 1: Misclassification Results

	0->1	1->0	total error
FOM (linear)	544 7.87%	174 2.52%	718 10.39%
Para-metric	171 2.47%	36 0.52%	207 2.99%
Logistic	167 2.42%	51 0.738%	218 3.15%

The nonlinear approaches have less than a third of the error of the conventional technique, and about twice the likely minimum error (i.e., all three techniques misclassified 107 (1.55%) of the cases). When the linear FOM judgments are employed, the model for the sensor environment appears as in Figure 7a, where the ellipses represent a given (high) confidence interval for the location of the emitter (small ellipses are best). Use of the parametric model leads to the more precise representation of Figure 7b.

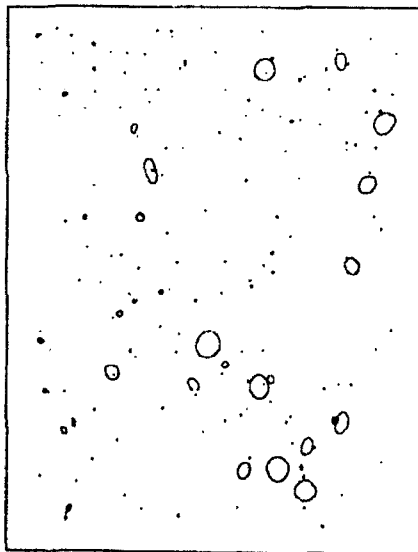


Figure 7a: FOM Environment Model

4. Conclusions

Visual analysis of the data provided a clear indication of the source of the problems with the current linear

discriminant approach to data correlation for emitter identification. The data show that linear discriminant approaches to this problem cannot easily handle the nonlinearities in the data, particularly due to the distance information. (10.4% of the pairs were misclassified.) Parametric approaches (3.0% error), and logistic regression (3.2% error) did however, exhibit positive performance. In fact, it was also determined that several other inductive methods, including k-nearest neighbors, kernel estimation, and neural networks, also performed well. Hence, we can build data correlation algorithms with improved performance using any of a variety of *nonlinear* approaches.

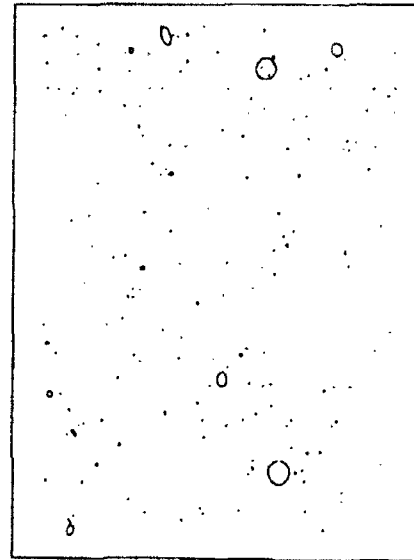


Figure 7b: Parametric Environment Model

References

- Barker, A., D.E. Brown, and W.N. Martin (1991). "A Framework for the Compact Representation of Data Used by Data Fusion Algorithms," Institute for Parallel Computation, School of Eng'g and Applied Science, University of Virginia, IPC-TR-91-005, May.
- Brown, D.E., C.L. Pittard, and A.R. Spillane (1992). "ASSET: A Simulation Test Bed for Evaluating Data Association Algorithms," *Computers and Operations Research*, forthcoming.
- Spillane, A.R., C.L. Pittard, and D.E. Brown (1989). "A Method for the Evaluation of Correlation Algorithms," Institute for Parallel Computation, School of Eng'g and Applied Science, University of Virginia, IPC-TR-89-004, April 1.
- Wright, F.L. (1980). "The Fusion of Multisensor Data," *Signal*, October: 39-43.

Assisted Visual Search for Data Structure

John F. Elder IV
 Institute for Parallel Computation &
 Department of Systems Engineering
 University of Virginia
 Charlottesville, VA 22903-2442
 (804) 982-2073 jfe7w@Virginia.edu

Abstract

A system is proposed to help analysis discover patterns in high-dimensional data. Projection pursuit and inductive modeling techniques are reviewed, with an intent toward their combination in an interactive graphic environment to effectively extend the scope of one's judgement a few dimensions. The proposed Inductive Data Exploration Algorithm (IDEA) will help statisticians to better build models from sample data by enhancing the process of "pondering point clouds".

1. Introduction

In the low dimensions of everyday experience, the human ability to rapidly recognize patterns in noisy data will likely never be matched by automata. However, the speed and persistence of computers must be harnessed to explore spaces of high dimension. Projection Pursuit (PP) algorithms (Friedman and Tukey, 1974) seek low-dimensional views of the data which score well by some index of "interestingness". When striving to model the data with PP regression (PPR; Friedman and Stuetzle, 1981), this perceived structure is iteratively removed and the filtered data is resubmitted, until only noise apparently remains.

The power of this idea is weakened in practice by several factors. Though PP was inspired by early graphical data manipulation systems, most PPR-like stagewise induction programs are run in batch mode, excluding potential mid-course contributions by an analyst. (Such induction programs include the Group Method of Data Handling, *GMDH* (Ivakhenko, 1968; Farlow, 1984); Classification and Regression Trees, *CART* (Breiman et al., 1984), the Algorithm for Synthesis of Polynomial Networks, *ASPN* (Elder, 1985); and Multiple Adaptive Regression Splines, *MARS* (Friedman, 1988).) PPR programs explore a single index (seeking unusual entropy, clustering, or skewness, etc.), and any structure found is addressed by a family of model components (e.g., splines). However, many types of patterns (with different properties and shapes) may prove interesting, so one could instead simultaneously unleash several projection index searches, as well as employ an arsenal of *corrective* basis functions (e.g., polynomial, threshold, and logistic elements).

In addition to expanding PPR goals and tools, both the interpretability of projections and the speed and quality of the global search can be enhanced. The proposed Inductive Data Exploration Algorithm (IDEA) combines

- 1) interactive graphics (including 3-D rotation, brushing, and plot linking) able to rapidly reveal low-dimensional structure,
- 2) automated statistical induction algorithms capable of discovering much of the structure of noisy data in high dimensions, and
- 3) information-theoretic criteria able to regulate the complexity of model approximations.

Still, it is anticipated that perhaps a chief strength of the algorithm will be its step "backwards" to critical reliance on the *analyst* in the structure search and removal process.

2. Inductive Modeling Overview

The goal of inductive modeling is to efficiently and robustly model high-dimensional data. As shown in Figure 1, given any two components of the set {Inputs, System State, Outputs} induction can be employed to estimate the third. For example, a *training* data base of inputs, X , and system states, S , can be employed to estimate the ensuing outputs, Y , (for unseen cases) in a *forecast* or prediction model. Likewise, it is a *design* problem to obtain a system, S , for given inputs and outputs X and Y -- and a *control* task to seek the best X for a given S and Y .

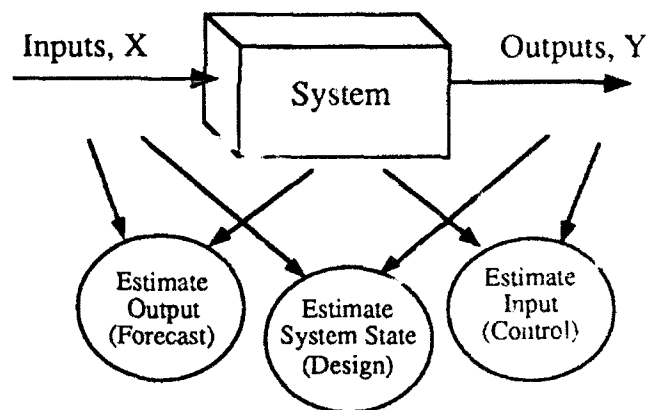


Figure 1: Three Types of Inductive Models

There are two broad classes of inductive techniques:

- 1) *Consensus* methods, which parametrically summarize data points, and
- 2) *Contributory* methods¹ which retain the training data and may employ one or every training case in the estimate for a new location in input space.

With the former, the training data are employed only to obtain a model (then discarded); in the latter, at least a subset of the data are retained for operation of the model. A brief overview of representative inductive techniques follows.

By far the most popular consensus technique is linear (least-squares) regression (LR). Usually, all the terms (i.e., the full model structure) are specified by the analyst and only the parameters are estimated by the computer. This is also the case for conventional artificial neural networks (ANNs); however, the nonlinear effect of changes in the ANN parameters force use of a much slower search technique (e.g., the stochastic gradient descent of *backpropagation*) than the closed-form matrix inversion of LR. Combining these two approaches, networks of regression nodes are employed in the family of GMDH techniques including the Polynomial Network Training Algorithm, PNETTR (Barron et al., 1984) and ASPN. A significant strength of those techniques however, is their allowing the model structure to grow out of the data, rather than being set by an analyst beforehand. Other such adaptive inductive tools include: CART, which employs piecewise constants for estimation; *Hinging Hyperplanes* (Breiman, 1991), using sums of multi-dimensional ramp functions; and *Class* (Cellucci and Hess, 1990), employing logistic regression elements.

The estimation surfaces formed by polynomial and sigmoidal methods are potentially quite nonlinear, although smooth. Spline techniques, like MARS, can adaptively construct similar surfaces (with a few continuous derivatives) which typically demonstrate more local response to "bumps" in the data. That is, a polynomial equation is *global*; a change to affect it one place, affects it everywhere (to varying degrees). A piecewise collection of splines however, can be adjusted more locally.

Contributory techniques embody this property of *local effects*. A classic example is the *nearest neighbor* (NN) algorithm, wherein the output for a new case is estimated to be the output of the known case closest by some measure (e.g., Euclidean distance) in input space. (The *k*-NN method just combines, e.g., averages, the outputs of the *k* nearest cases.) If closer neighbors are to be weighed more heavily

than distant ones, then *kernel* estimation (Parzen 1962) is appropriate. There, a kernel function, $h(\cdot)$ (such as a rectangle, triangle, or gaussian node) of a given width is centered over each known case, i , and is evaluated at the location of interest (the new case, j) to supply the relative weight, w_i :

$$Y_j \approx \frac{\sum w_i Y_i}{\sum w_i}, \quad i = 1..N$$

for N training cases, where $w_i = h(\|X_i - X_j\|)$. (A rectangular kernel is equivalent to a *k*-NN approach, with k depending on location.) In classification, the relative sum of weights of each class at a location provides an estimate of its conditional probability; and, in density estimation, it forms the density surface itself. This last is analogous to dribbling unit piles of sand (for 2-d gaussian kernels) over each case location to build up the density estimation surface. Kernels lead therefore, to a type of *histogram* with "bins" following the data, rather than forcing data into pre-established bins. Indeed, *averaged shifted histogram* techniques have been employed (Scott, 1985) which have properties intermediate between the smoothness of kernels and the speed of histograms.

Radial basis functions (RBFs) are a contributory technique with a consensus aspect. Radially-symmetric functions (similar to kernels, although they may increase with distance, if desired) are centered over each training case. Yet, instead of being used directly (as with kernels) these w_i values become variables in an LR data base. That is, each training output, Y_j , is modelled as a weighted sum of a constant and the $N-1$ distance functions $h(\|X_i - X_j\|)$, $i \neq j$. As the number of constraints, N , matches the degrees of freedom available, there is an exact model (zero training error) which is used to interpolate new cases. In practice though, the variables are often too numerous for model robustness, and the training data is sub-sampled (e.g., Chen, Cowan, and Grant, 1991); that is, features are removed to reduce collinearity (e.g., Elder, 1990).

Wavelets are a contributory technique receiving increasing attention due to useful theoretical properties and their ability to describe functions with rapid level transitions (see, for example, the Bock, Chui, Johnstone, or Walter papers in these proceedings). Another recent estimation method can be taken from these proceedings as well: *Delaunay planes*. In (Elder, 1992) the expected value of a stochastic function, known only at a set of training locations, is represented by a piecewise planar surface, where each plane intersects a simplex of points defined as a set according to the "empty circumsphere" rule of Delaunay triangulation. The variance of this estimate (assuming a random field, or fractal surface) is gaussian, and grows from a minimum at the known points to peak at uncharted

¹Contributory methods are most often referred to as *nonparametric* techniques, though some parameters are usually involved; e.g., bin or kernel widths, number of neighbors, type of wavelet, etc.

locations -- forming a piecewise quadratic "variance canopy" which arches over the expectation planes. Though designed for a global optimization algorithm, this method could also be employed directly for estimation.

A wide range of algorithms for automatically inducing models from sample data are emerging in this rapidly growing field. (A sample of further promising methods includes *locally weighted regression* (or *loess*; e.g., Cleveland and Devlin, 1988); *stochastic* models (Sacks, Schiller and Welch, 1989); *slicing inverse* regression (Duan and Li, 1991); *Fit-Short* (an adaptive kernel method; Kozek and Schuster, 1991), and *composite* models (Skeppstedt, Ljung and Millnert, 1992).) Most of the techniques have distinct strengths, especially in the (often implicit) assumptions made about the data, and in the requirements made of (or opportunities available to) the user to direct the analysis. An environment that allowed a subset of these (semi-) automatic methods to augment visual search (as discussed below), should prove quite useful.

3. Induction Hierarchy

To help classify the growing list of viable induction methods, a hierarchy of induction is proposed, scaled by the increasing degree to which modeling "decisions" are left to the computer:

- 1) *Heureka*: The analyst specifies the entire model. (Like the legend of Athena from Zeus, it springs forth fully formed from one's head -- terms, weights, and all.)
- 2) *Knobby*: One specifies the structure (e.g., inputs and terms) and the computer searches for parameter settings (weights) which best fit the data. This is the most common category of models, including LR and ANNs.
- 3) *Plastic*: The analyst specifies the model class (polynomials, for example) and initiates a search for both structure and weights. This class can be further subdivided into:
 - a) *Stretchable*. The structure can grow in only one dimension (say, the order of a polynomial in one variable); so, the decision concerns the best cutoff point. Further examples include determining the number of useful terms of a power series expansion, or the frequencies of a low-pass spectral approximation.
 - b) *Accretable/Erodible*. The structure can grow and/or diminish in several dimensions. Ex: Adaptive regression techniques, such as "greedy" forward term selection (add that term which most helps), greedy reverse term removal (cut the term least aiding the model), and their step-wise combination.

- c) *Malleable*. The model is a composition of functions drawn from a given family (e.g., GMDH polynomials); thus, the form is very flexible and is capable of growing in large *chunks*. Thereby, the short-term optimality of greedy growth is extended more to the medium-term. Malleable techniques are capable of building new candidate features when lower-order versions of those features prove useful.

4. *General*: The analyst provides little more than representative data, and the algorithm searches across model families (sets of basis functions) and perhaps even across error metrics. No implementation of such an ambitious method is known, though the Minimum Description Length (MDL) model selection criterion (Rissanen, 1978) and other recent means of complexity regulation (A. Barron, 1991; Faraway, 1991) may be capable of judging between candidate models generated by such a search procedure.

A further distinction can be made between *closed* techniques, for which the library of potential terms must be specified by the analyst beforehand, and *open* methods, which effectively expand their libraries of potential terms as the model grows. The former includes branch and bound regression (e.g., Furnival and Wilson, 1974), which could be labelled a "reverse erodible plastic method" optimal for its closed library of polynomial terms. Along with techniques in the GMDH family, CART provides an example of open (plastic) modelling. During synthesis of its decision tree, additional bifurcations are considered for each impure leaf (i.e., each currently terminal node with more than one class or estimation value held within it.) As the tree grows (or is pruned in a later simplifying stage), the list of input space partitions (the "terms" of the model) changes, so the CART term library can be considered "open".²

4. Visualization Helps

The trend in the above hierarchy is to turn over more and more of the modelling process to our untiring, precise, obedient³, blindly fast (but dimwitted) assistant, the computer. But what does our trusty aide perceive? Typical algorithms key on summary statistics of the data being modelled. However, Anscomb (1973) showed that, in terms of LR error statistics, data sequences as diverse as

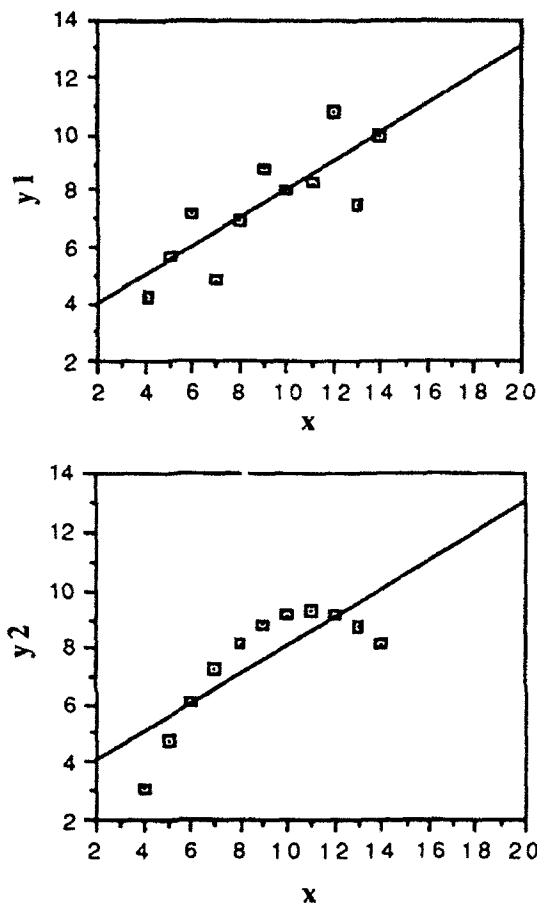
²With univariate thresholds (the CART default), potential cuts are a subset of the power set of threshold functions of the training data, and so are a closed set. Though the greedy accretion and erosion algorithm of CART only considers a fraction of the $O(N^k)$ potential terms for N cases of k variables, the number is large enough to be effectively open.

³"I really hate this darn machine;

I wish that they would sell it.

It never does quite what I mean, but only what I tell it."

those of Figure 2 can appear identical. Surprisingly, those graphs have the same mean squared error ($MSE=1.25$), coefficient of determination ($R^2=.67$), and linear fit ($y = .5x + 3$). Only for y_1 is the LR assumption of gaussian errors about a line upheld; the data of y_2 are instead quadratic, an



outlier upsets the trend of y_3 , and an *influential* (non-outlying) case drives the estimator for y_4 . A "stretchable" inductive method could correctly specify the quadratic structure of y_2 , but *case analysis* (e.g., Belsley, Kuh, and Welsh, 1980) is required to discern problems with the latter two -- something not yet built into most induction methods.

Of course, visual representation of low-dimensional problems (as in the Figure), make the situation clear in a way statistical summaries cannot, and suggest remedies that might otherwise go unnoticed. An instructive experience for this author occurred (years ago) on a data set eventually revealed to be similar to that of y_3 , but with the outlier (due to a recording error) on one end. An otherwise powerful inductive algorithm chose a high-order model to fit the data, having no explicit ability to flag or down-weight the outlying case. However, the algorithm was able to winnow away a mass of variables confusing the picture and identify the one essential data feature. Plotting then revealed the

(unexpected) simplicity of the relationship, marred only by an outlier -- the cause of which was quickly tracked down. This paper, in essence, merely attempts to outline a formal joining of such automatic and interactive tools.

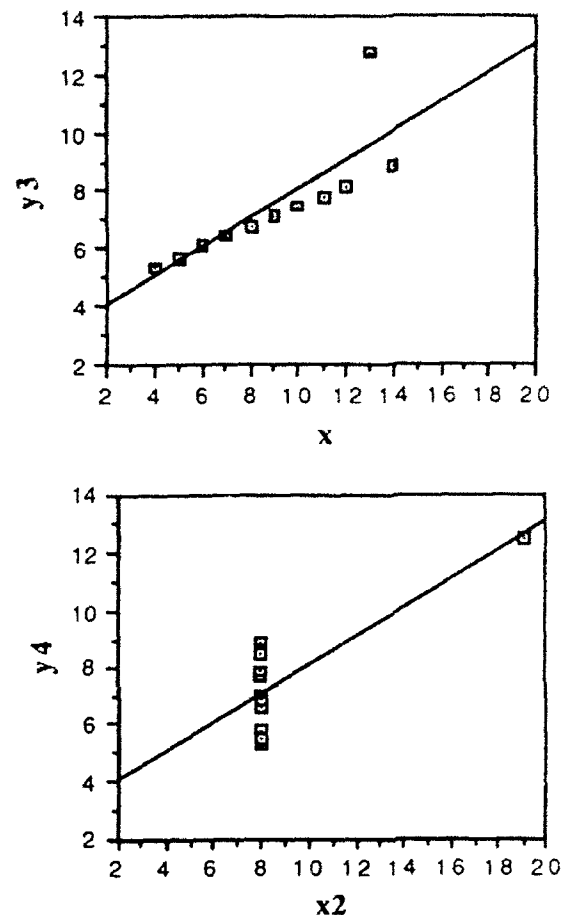


Figure 2: Anscomb's Quartet

5. Projection Pursuit

In high dimensions, visual strategies include *plot matrices*, a "grand tour" (Asimov, 1985), and projection pursuit (PP). Though a plot matrix, or pairwise arrangement of scatterplots, is only a set of two-dimensional views of the data, this is often sufficient to provide useful insight. Also, graphic programs are increasingly enforcing case links between graphs (through different symbols and colors, *brushing*, etc.; e.g., Cleveland and McGill, 1988) -- extending the effective dimension of perception. The grand tour strategy presents the analyst with a single 2- or 3-dimensional projection of the data where the view (projection matrix) smoothly transitions in such a way that the analyst is eventually presented with "all sides" of the point cloud. Interesting views are meant to be saved for later analysis, or to cause the tour to be postponed for local exploration.

In high dimensions, a grand tour can take overlong, so it is useful to have the computer score candidate views (noting low entropy, high clustering, unusual skewness or Fisher information, etc.), and present, or steer the tour through, only the best views. This is projection pursuit, and such a PP tour has recently been implemented in the workstation package, *XGobi* (Swayne, Cook, and Buja, 1991). The PP index could also include factors such as *interpretability* (Morton, 1989) complexity, or novelty -- the choice might depend critically on the application. In Figure 3, a 2-d point cloud of two classes is plotted, surrounded by some (histogram) projections of the data. The index (to be minimized over 180° of θ) is the #misclassifications from the best θ threshold for that view.

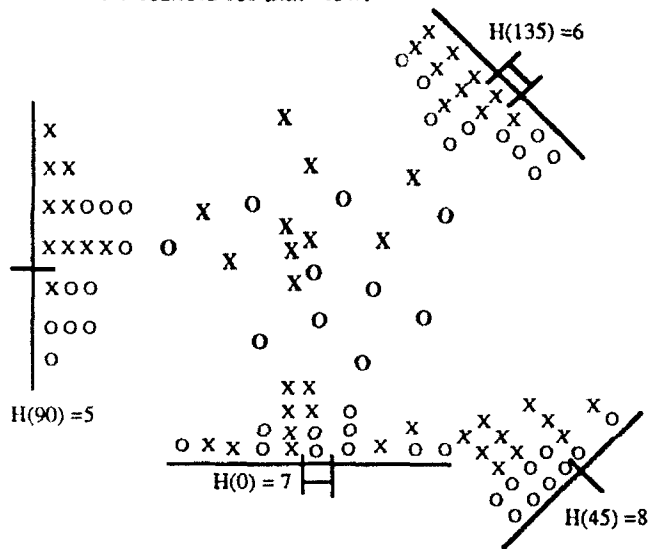


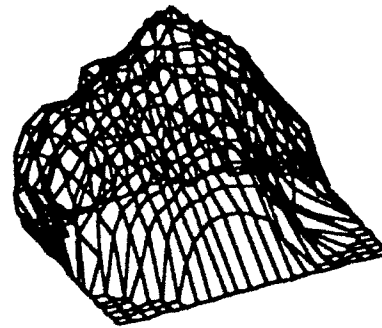
Figure 3: 1-Dimensional Views of 2-d Data

Note that the search space wraps around ($\theta=180+\theta$), causing a redundancy which could save search time if exploited (though its complexity has apparently prevented that, thus far). For example, when reducing 3-d data to a 1- or 2-d view, the projection axis can be represented as a point on a unit hemisphere, as in Figures 4a and 4b, where the "score" for a location is represented by its distance from the center of the sphere. However, for search continuity, the most distant points on the base of the dome should actually be connected (a 3-d wrap-around). As indicated by the Figures, projection spaces can be quite *rough*; they also can vary in shape widely due to changes in the projection index (the score function) (e.g., Cook, Buja and Cabrera, these proceedings). Therefore, a sophisticated global (not local) search is required to discover and rank the projections to be presented to the analyst.

6. Summary

Automated induction and projection pursuit algorithms, despite their variety, are not (yet?) competitive with (even

beginning) analysts in perceiving structure in noisy data -- in the low dimensions of human experience. Of course, in high-dimensional applications, where the structure of the underlying "data generating machinery" is unclear, automated induction algorithms are quite useful. Often through nothing more than trial and error, they can construct compositions of small functions into empirical approximations of a function interpolating (and smoothing) the training data. However, automated methods must handle a host of data hazards, including influential and outlying cases, and *collinear* and *redundant* variables. Most importantly, empirical methods must avoid *overfit*; that is, the tendency to fit the noise, as well as the signal, in the data.⁴ Also, the typical summary statistics monitored by most algorithms do not always reveal key data relations.



Figures 4a,b: 3-Dimensional PP Surface Views

Fortunately, sophisticated interactive graphic environments are becoming more widespread. A system is envisioned wherein such an environment would form the base, and would be augmented by a variety of automated algorithms of two forms: *structure exploration* (like PP)

⁴Performance is monotonic with activity; that is, everything improves the training accuracy of the model. One must rely on other data (e.g., GMDH "checking set"), training subsets (cross-validation), or a complexity penalty (such as MDL).

and structure removal (i.e., approximation). That is, in the background⁵ of one's graphical manipulations, the computer could continually search for, and queue up, a list of low-dimensional views of the data which score well by any of a number of PP criteria. Then, when a significant bump or trend (or blip or ...) is found, a variety of approximative methods could be employed (kernels, polynomials, etc.) to fit it. The model could then be removed from the data (leaving a residual, and requiring an update of all extant views) and the process repeated until only noise appears to remain. (To avoid overfit, it is anticipated that a complexity budget even more strict than usual must be adhered to, because of the increased approximation flexibility.)

In this manner, perhaps the muddled view the computer has of high dimensions can be enough to lead us "walking pattern recognizers" to the select low dimensional views in which all is made clear.

References

- Anscomb, F.J. (1973). Graphs in Statistical Analysis, *American Statistician* 27: 17-21.
- Asimov, D. (1985). The Grand Tour: A Tool for Viewing Multidimensional Data. *SIAM J. on Scientific and Statistical Computing* 6: 128-143.
- Barron, A.R. (1991). Complexity Regularization with Application to Artificial Neural Networks, in *Nonparametric Functional Estimation and Related Topics*, G. Roussas (ed.), Kluwer, Netherlands: 561-576.
- Barron, R.L., A.N. Mucciardi, F.J. Cook, J.N. Craig, A.R. Barron (1984). Adaptive Learning Networks: Development and Application in the United States of Algorithms Related to GMDH, Ch. 2 in *Self-Organizing Methods in Modeling: GMDH Type Algorithms*. S.J. Farlow (ed.), Marcel Dekker, New York.
- Belsley, D.A., E. Kuh, R.E. Welsch (1980). *Regression Diagnostics: Identifying Influential Data and Sources of Collinearity*. Wiley, New York.
- Breiman, L. (1991). Hinging Hyperplanes for Regression, Classification, and Function Approximation, *Tech. Rpt. 324*, Dept. Statistics, UC Berkeley, CA.
- Breiman, L., J.H. Friedman, R.A. Olshen, C.J. Stone (1984). *Classification and Regression Trees*. Wadsworth & Brooks, Monterey, California.
- Cellucci, R.L., P. Hess (1990). Techniques for Developing and Applying Polynomial Network Synthesis Software, *Proc. 22nd Symposium on the Interface: Computing Science and Statistics*, East Lansing, MI, April.
- Chen, S., C.F.N. Cowan, P.M. Grant (1991). Orthogonal Least Squares Learning Algorithm for Radial Basis Function Networks, *IEEE Trans. on Neural Networks* 2, no. 2: 302-309.
- Cleveland, W.S., S.J. Devlin (1988). Locally Weighted Regression: An Approach to Regression Analysis by Local Fitting, *J. Amer. Stat. Assoc.* 83 #403: 596-610.
- Cleveland, W.S., M.E. McGill (1988). *Dynamic Graphics for Statistics*. Wadsworth, Inc., CA.
- Duan, N., K.C. Li (1991). Slicing Regression: A Link-Free Regression Method, *Ann. Stat.* 19 no. 2: 505-530.
- Elder, J.F. IV (1985). *User's Manual: ASPN: Algorithm for Synthesis of Polynomial Networks* (4th Ed., 1988). Barron Associates Inc., Stanardsville, Virginia.
- Elder, J.F. IV (1990). Feature Elimination Using 'High-Order Correlation', *Proc. Aerospace Applications of Artificial Intelligence*, Dayton, OH, Oct. 29-31 p. 65-72.
- Elder, J.F. IV (1992). A Novel Efficient Global Search Algorithm for Multiple Dimensions, *Proc. 24th Symposium on the Interface: Computing Science and Statistics*, College Station, Texas, March 18-21.
- Faraway, J.J. (1991). *On the Cost of Data Analysis*, Tech. Rpt. 199, Dept. Statistics, Univ. Michigan, Ann Arbor.
- Farlow, S.J. (1984), Ed. *Self-Organizing Methods in Modeling: GMDH Type Algorithms*. Marcel Dekker, NY.
- Friedman, J.H., J.W. Tukey (1974). A Projection Pursuit Algorithm for Exploratory Data Analysis, *IEEE Trans. Computers* 23: 881-889.
- Friedman, J.H., W. Stuetzle (1981). Projection Pursuit Regression, *J. of the American Statistical Assoc.* 76, no. 376: 817-823.
- Friedman, J.H. (1988). Fitting Functions to Noisy Scattered Data in High Dimensions, *Proc. Computing Science & Statistics: 20th Symposium on the Interface*.
- Furnival, G.M., R.W. Wilson, Jr. (1974). Regressions by Leaps and Bounds, *Technometrics* 16: 499-511.
- Ivakhnenko, A.G. (1968). The Group Method of Data Handling -- A Rival of the Method of Stochastic Approximation, *Soviet Automatic Control* 3.
- Kozek, A.S., E.F. Schuster (1991). On 'Fit the Short Curve' Principle for Smoothing Nonparametric Estimators, *Proc. 23rd Symposium on the Interface: Computing Science and Statistics*.
- Morton, S.C. (1989). *Interpretable Projection Pursuit*, Tech. Report 106, Dept. Statistics, Stanford Univ., CA.
- Parzen, E. (1962). On Estimation of a Probability Density Function and Mode, *Annals Math. Stat.* 33: 1065-1076.
- Rissanen, J. (1978). Modeling by Shortest Data Description, *Automatica* 14: 465-471.
- Sacks, J., S.B. Schiller, W.J. Welch (1989). Designs for Computer Experiments, *Technometrics* 31 no. 1: 41-47.
- Scott, D.W. (1985). Averaged Shifted Histograms: Effective Nonparametric Density Estimators in Several Dimensions, *Ann. Statist.* (I think).
- Skeppstedt, A., L.Ljung, M. Millnert (1992). Construction of Composite Models from Observed Data, *Int. J. Control* 55 no. 1: 141-152.
- Swayne, D.E., D. Cook, A. Buja (1991). *A User's Manual for XGobi, a Dynamic Graphics Program for Data Analysis Implemented in the X Window System* (Release 2), Bellcore.

⁵My computers spend most of their time waiting for me...

Classification Procedures and its Application

R. K. JAIN

*Department of Mathematics and Statistics
Memorial University of Newfoundland
St. John's, Newfoundland, Canada A1C 5S7*

Abstract

Discriminant procedures are often used to classify data based on observed characteristics of the response variables. This paper discusses the validation techniques in the use of discriminant function approach. Numerical example is used to illustrate its application.

1 Introduction

Numerous classification procedures exist in statistical literature. For example, some of the classification procedures include likelihood ratio test [Anderson (1958)], information theory [Kullback (1959)] and Bayesian techniques [Geisser (1964 & 1965)]. However, in this paper, we focus our attention to discriminant function approach suggested by Fisher (1936), for classifying data based on various characteristics of the response variable. The choice of method for classifying the observation depend on the nature of the data. If the data are multivariate normal and the covariance matrices are not too far apart, then the linear discriminant function approach can be used. However, if the covariance matrices differ unduly, the quadratic discriminant function can be used. Marks and Dunn (1974) discussed the performance of these discriminant functions (viz., Fisher's linear discriminant function, the best linear discriminant function and quadratic discriminant function) in classifying observations into two groups when covariance matrices are unequal, based on Monte Carlo studies. The per-

formance was compared based on overall probability of misclassification. A review of a 'best' linear discriminant function is described by Marks and Dunn (1974). Both the quadratic and best linear discriminant function are Fisher's linear function if covariance matrices are equal.

Seber (1984) discussed many simulation studies to make comparison between linear discriminant function and quadratic discriminant function approach. The problem with quadratic discriminant function is that the large numbers of parameters to be estimated leads to unstable estimates. Dillon (1979) reviewed the performance of the linear discriminant function in situations where the assumption of equality of covariance matrices is violated. He also discussed a procedure, in which the total sample is split into two subsamples: one subsample is used to construct the discriminant function and the other subsample is used for validation. The method was evaluated based on misclassification error rates. The assessment of predictive accuracy in discriminant analysis was further discussed by Huberty, et. al (1987). He examined the estimation technique of these error rates: optimal, actual and expected actual error rates. Monte Carlo sampling was used to compare the performance of various methods described by Huberty, et. al (1987).

The purpose of the present study is to examine the application of discriminant analysis procedure to medical data and evaluate the performance based on classification error rates. In section 2, the discriminant function approach to classify observations into various

groups is reviewed, the score used to classify a observation is described, and the performance statistics are defined. In Section 3, the medical data are described to illustrate the application. The statistics used in explaining the discriminant function approach based on misclassification rates are discussed. Finally, in Section 4, the importance of the various forms of training samples based on simulation studies is discussed.

2 A Brief Review of Discriminant Procedure

The discriminant analysis is used to develop a rule for classifying an observation (viz., status of disease) into one of G groups (i.g., well, moderate and severe, etc.) on the basis of p measured variables using a training sample with n cases. Suppose we have n_i observations \underline{x}_{ij} ($i = 1, 2, \dots, g, j = 1, 2, \dots, n_i$) from multivariate normal distribution with $\underline{\mu}_i$ and covariance matrix Σ_i . Let

$$S = \sum_{i=1}^g (n_i - 1) S_i / n - g,$$

where

$$n = \sum_{i=1}^g n_i$$

and

$$S_i = \frac{\sum_{j=1}^{n_i} (\underline{x}_{ij} - \bar{\underline{x}}_i)(\underline{x}_{ij} - \bar{\underline{x}}_i)'}{n_i - 1}$$

be the pooled estimate of Σ . The linear discriminant function is given by

$$L_i(\underline{x}) = \log \pi_i + \bar{\underline{x}}_i' S^{-1} (\underline{x} - \frac{1}{2} \bar{\underline{x}}_i). \quad (1)$$

A detailed description of assigning observations to various groups are discussed in Seber (1984).

Probability of Misclassification

Let π_i be the proportion of population in group G_i ($i = 1, 2, \dots, g$) such that $\sum_{i=1}^g \pi_i = 1$. Let $f_i(\underline{x})$ be the

probability density function of \underline{x} if $\underline{x} \in G_i$. The probability of misclassifying an observation of G_i is

$$P(i) = \sum_{j=1, j \neq i}^g P(j/i) \quad (2)$$

where

$$P(j/i) = \int_{R_j} f_i(\underline{x}) d\underline{x},$$

and $\{R_1, R_2, \dots, R_g\}$ is a partition of the sample space R such that a member of population assign to G_i if $\underline{x} \in R_i$.

The total probability of misclassification is $\sum_{i=1}^g \pi_i P(i)$. The optimum assignment rule based on minimizing the misclassification probability is discussed by Seber (1984). A test of multivariate of normality and equal covariance matrices is given by Hawkins (1981). When the covariance matrices are unequal, the quadratic discriminant function approach should be used.

3 Application

The data for this study include 450 cases, classified as types IIa, IIb, and IV called hyperlipoproteinemias, and normal. The following variables are measured for each individual case: high-density lipoprotein, total cholesterol, triglycerides, low-density lipoprotein, and pseudocholinesterase. BMDP7M computer programme is used to randomly splitting the sample and cross-validating the classification function. A series of simulated training samples based on sample size of 10, 20, 50, 60, 70, and 80 percent of the total sample are obtained to investigate the performance of linear discriminant function. 40 training samples are generated for each set of combinations. Let \bar{p} be the estimated proportion of correct classification based on 40 training samples generated for each individual combination. Table 3.1 presents the results of estimated proportion of correct classification based on cross-validation over the held-out sample.

Table 3.1. Estimated Proportion of Correct Classification.

Percentage of Training Sample	\bar{p}	Standard Deviation of \bar{p}
10	0.7444	0.0689
20	0.7631	0.0672
30	0.7701	0.0666
40	0.7856	0.0654
50	0.7948	0.0638
60	0.7967	0.0636
70	0.7989	0.0634
80	0.8035	0.0629

Concluding Remarks

Jain et. al (1983) used the classification procedure to classify hyperlipoproteinimias on the basis of the concentrations of cholesterol, triglycerides, and pseudo-cholinesterase, etc. Identifying the significance and independence of various patients characteristics related to disease-process is vital. This paper examined how large the training sample should be to ensure that the discriminant function approach is applicable. Table 3.1 indicates that the estimate of proportion of correct classification stabilizes as the sample size increases. It seems that splitting sample 50-50 would give the desirable results. However, further research work for different sets of medical data is under investigation to set some sort of guidelines in choosing the training sample.

References

1. Anderson, T. W. (1958). *An Introduction to Multivariate Statistical Analysis*. Wiley.
2. Dillon, W. R. (1979). the performance of the linear discriminant function in nonoptimal situations and the estimation of classification error rates: A review of recent findings. *Journal of Marketing Research*, 16, p. 370-381.
3. Fisher, R. A. (1936). The use of multiple measurements in taxonomic problems. *Annals of Eugenics*, 7, p. 179-188.
4. Geisser, S. (1964). Posterior odds for multivariate normal classification. *Journal of the Royal Statistical Society, Series B*, 26, p. 69-76.
5. Geisser, S. (1965). Bayesian estimation in multivariate analysis. *Annals of Mathematical Statistics*, 36, p. 150-159.
6. Hawkins, D. M. (1981). A new test for multivariate normality and homoscedasticity. *Technometrics*, 23, p. 105-110.
7. Huberty, C. J., Wisenbaker, J. M., and Smith, J. C. (1987). Assessing predictive accuracy in discriminant analysis. *Multivariate Behavioral Research*, 22, p. 307-329.
8. Jain, R. K., Kutty, K. M., Huang, S., and Kean, K. (1983). Pseudo-cholinesterase/high-density lipoprotein cholesterol ratio in serum of normal persons and of Hyperlipoproteinemics. *Clinical Chemistry*, 29, p. 1031-1033.
9. Kullback, S. (1959). *Information Theory and Statistics*. Wiley.
10. Marks, S., and Dunn, O. J. (1974). Discriminant functions when covariance matrices are unequal. *Journal of the American Statistical Association*, 69, p. 555-559.
11. Seber, G. A. F. (1984). *Multivariate Observation*. Wiley.

The Application of Kriging for the Controlled Minimization of Large Data Sets

David Robinson, Christopher Brodtkin

Dept of Aeronautics and Astronautics

Air Force Institute of Technology

Wright-Patterson AFB, OH 45433 *

1. Introduction

Data collection methods are quickly outstripping the ability of current computer analysis and visualization routines. In addition to limitations in processing capabilities, the user is obviously being overwhelmed with information.

Recent advances in medical data collection have provided users with vast amount of data in the form of CAT, MRI, and PET scans. Satellite image analysis has become more important with the need for treaty verification and the increased role of third world nations in global politics. The technological breakthroughs in rapid prototyping coupled with the expanding capabilities of computer-aided design tools and data collection methods have given the design engineer the capability to go from concept to production in a matter of weeks.

However, all of these advances have one problem in common: vast amounts of data that needs to be reduced for analysis with limited loss of accuracy. The problem is not one of data *compression*, but rather data *reduction*: removal of selected data points from the original data set while preserving the integrity of the data. Current data compression methods, while reducing the amount of data, in terms of total bytes, are inherently lossy. This is acceptable when only the visual characteristics are important, but when dealing with medical images or computer-aided design environments, this loss of information and accuracy is not acceptable.

2. Approach

It is desired to represent the surface S with a reduced number of points such that the resulting mean squared error associated with the new surface can be controlled by the analyst. Let x_1, \dots, x_N denote a complete set of points available to describe the surface $S \subset \mathcal{R}^3$. Assume that we wish to describe this surface with a maximum of $n \leq N$ points and that an initial subset of points

has been selected: x_1, \dots, x_k ($k < n$). The selection of the final $n - k$ points is based upon the solution to the constrained optimization problem: [ref Sydarovski]

$$\begin{aligned} &\text{Minimize:} && n - k \\ &\text{subject to:} && \max_Z [Var(Z_0 - Z)] \leq \epsilon \end{aligned}$$

where ϵ is the maximum allowable estimation error variance across the surface composed of the points: $(x_1, \dots, x_n, t_1, \dots, t_{n-k})$. Note that the dependence of the estimation variance on the number of samples n is explicitly included here.

A number of techniques can be employed to determine the optimal set of additional points t_1, \dots, t_{n-k} including exhaustive search or branch and bound but generally at a high computational price. Alternatively, methods such as sequential inclusion suggested by Szidarovsky (1983), while sub-optimal, provide an attractive compromise. However, for the size of the data set investigated here ($N > 71,000$), this is still computationally intensive. A simplification of this method has been employed that adds points where the largest immediate reduction of the estimation error variance can be achieved.

The data reduction approach outlined here accounts for local redundancies in the data, and specifically characterizes those redundancies with a spatial variogram function:

$$\gamma(h_{ij}) = \frac{1}{2} Var\{Z(x_i) - Z(x_j)\}$$

where h_{ij} is the vector distance between the points $x_i = (X_i, Y_i)$ and $x_j = (X_j, Y_j)$, h_{0j} is the distance between the point to be estimated, x_0 , and the known point, x_j , and $Var(\cdot)$ is the statistical variance. For the applications to be discussed here (medical and satellite imaging), the value x_i will represent the two-dimensional location of a particular point and $Z_i = z(x_i)$ will represent the grey-scale value at that location; extension to three-dimensions is straight forward.

It is desired to represent to estimate the surface Z_0 at location (X_0, Y_0) using an optimal linear predictor of the form: $Z_0 = \sum_{i=1}^n w_i Z_i$. Given an estimate for the

*This work was performed under the sponsorship of Armstrong Laboratory, Human Engineering Division, Wright-Patterson AFB, OH 45433

correlation structure of the image and a sample from the original surface, the best linear unbiased prediction of the complete surface is found by the simultaneously solving the set of linear equations:

$$\zeta = A^{-1}\beta$$

where:

$$\zeta = \{\lambda, \alpha_1, \alpha_2, w_1, w_2, \dots, w_n\}^T$$

$$\beta = \{1, X_0, Y_0, \gamma(h_{10}), \gamma(h_{20}), \dots, \gamma(h_{n0})\}^T$$

and:

$$A = \begin{bmatrix} 0 & 0 & 0 & 1 & 1 & \dots & 1 \\ 0 & 0 & 0 & X_1 & X_2 & \dots & X_n \\ 0 & 0 & 0 & Y_1 & Y_2 & \dots & Y_n \\ 1 & X_1 & Y_1 & \gamma(h_{11}) & \gamma(h_{12}) & \dots & \gamma(h_{1n}) \\ 1 & X_2 & Y_2 & \gamma(h_{21}) & \gamma(h_{22}) & \dots & \gamma(h_{2n}) \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & X_n & Y_n & \gamma(h_{n1}) & \gamma(h_{n2}) & \dots & \gamma(h_{nn}) \end{bmatrix}$$

The best estimate of the surface at point (X_0, Y_0) is then given by:

$$Z_0 = w_1 Z_1 + w_2 Z_2 + w_3 Z_3 + \dots + w_n Z_n$$

and the associated variance of the error between the estimated value Z_0 and the true value Z is:

$$\text{Var}(Z_0 - Z) = \sum_{i=1}^n w_i \gamma(h_{i0}) + \lambda + \alpha_1 X_0 + \alpha_2 Y_0$$

2.1. Variogram

As mentioned previously, the method proposed depends heavily on characterizing the local redundancies that exist in the surface. An estimate of the variogram function used to characterize these redundancies is found by using weighted-least squares to fit a function to the points:

$$\gamma^*(h) = \frac{1}{2|N|} \sum_{i=1}^{|N|} [z(x_i + h) - z(x_i)]^2$$

where $|N|$ is the number of pairs of data values at a distance of h apart from one another, x_i is the location of point i , $x_i + h$ is the location of a point at distance h from i , and $z(x_i)$ and $z(x_i + h)$ are the values of the quantity of interest at i and $x_i + h$. From experience, the spherical class of variogram models has been found to the most appropriate (and sufficiently robust) for the type of data analyzed. The general form of the spherical model:

$$\gamma(h) = \begin{cases} C(\frac{3}{2}\frac{kh}{a} - \frac{1}{2}\frac{(kh)^3}{a^3}) + C_0 & \text{if } h < a \\ C + C_0 & \text{if } h \geq a \\ 0 & \text{if } h = 0 \end{cases}$$

The spherical model is defined by three parameters: a , C , and C_0 . The first parameter, a , is referred to as the range and is used to determine the range of influence or neighborhood. The third parameter, C_0 , is known as the nugget effect, while the second parameter, C , is used in conjunction with C_0 to determine the sill, $(C + C_0)$. The parameter k is a scaling factor to account for any geometric anisotropy that may be present.

2.2. Selection of Minimal Sample Set

Two methods for minimization were developed. The first takes advantage of the lattice structure of the data to minimize the computational time. The second method is general in nature and can be applied to any multidistributed data set. Both methods are based upon achieving an estimation variance less than a given bound using a minimal number of observation points.

2.2.1 Lattice Data

In general, once the variogram is known, the variance of the estimation error at a point is a function only the relative distance and location of the known points in the neighborhood to the point to be estimated. If a 'pattern' of points could be chosen such that the largest variance is just within the maximum allowed (ϵ) then that pattern of points would constitute the optimal (minimal) data set.

When the original data has a lattice structure (as in most graphic images) the pattern selection of the data pattern is a matter of selecting a set of points such that the maximum distance between any pair of neighboring points in the pattern is just less than that which produces the maximum error allowed variance. If the points are closer than this distance then the data set may not be minimal. If the points are farther apart than this distance then the maximum error variance criteria won't be met.

This suggests that a rectangular pattern, due to geometric anisotropy, would produce an optimal minimal data set and this observation was verified in the final application. A typical rectangular pattern is shown in Figure 1 where \bullet is a known point in the data set and \circ represents points to be estimated. The location of the point with the largest estimation variance will be at the point that is farthest from all known points. For a rectangle this will be at the intersection of the diagonals.

Therefore, only that point at the intersection of the diagonals of the rectangular pattern needs to be considered to determine the maximum estimation variance within that kriging area. The routine starts with an initial sample of points chosen such that the distance between the center point and the corner points is an integer value just less than the variogram range. The distance

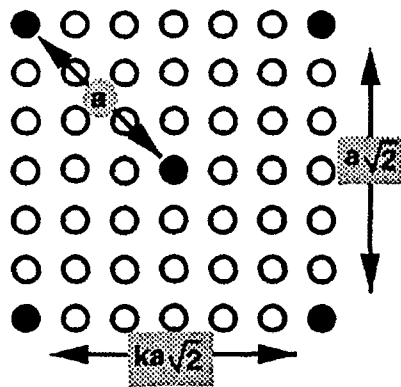


Figure 1: Lattice Data Neighborhood Representation

is then iteratively reduced until the estimation variance is just less than the maximum allowed variance set by the user.

3. Application and Results

Magnetic Resonance Image (MRI) scans were used to demonstrate the applicability of the method. The scans are comprised of several parallel planes of data in which each plane, or slice, is 268 measurements, (pixels) wide and 267 measurements, (pixels) long. Each measurement is an intensity represented as a grey scale value between 0 and 255. For this application the pixel location in each slice, row and column, are used as the coordinates of the point and the grey scale value is used as the quantity of interest at each point.

Due to the nature of the data no global trend was removed. It is important to note that, for this effort, the background of each image (noise and possibly support structure), which is normally filtered out by the display program, was considered to be known data in addition to the scan data from the specific image area associated with the brain. Table 1 summarizes the results for a particular MRI slice. The analysis was duplicated on over 100 slices on three different subjects with similar results. The magnitude of the data reduction indicated in Table 1 was extreme, in that typically much higher reductions were seen. However, the slice presented here contained the most detail and was the most sensitive to quality judgements.

Obviously the most important aspect of the image is the area containing information regarding the tissue density of the brain. It was noted previously that background noise and clutter that occurs during the MRI scan was also included in the results. This was done so that there could be no possibility of bias in the minimization procedure; no knowledge of the original image

Table 1: Slice Statistics

Figure	Largest Variance	Percent Reduction
2.a original data	0	0.0
2.b	42.9	50.0
2.c	50.2	87.36
2.d	57.1	94.34

was introduced.

However, by using a rather simple, automatic partitioning algorithm to first identify regions of the image where isotropic behavior was clearly present, and then minimizing each partition independently, a further reduction was always possible. This further reduction was dependent on the size of the 'interesting partition' relative to the background; as the background began to dominate the image, data reduction increased proportionately.

4. Conclusions

In conclusion, this paper develops and demonstrates the application of spatial statistics for the determination of an 'optimal' number of data to accurately define a complex surface. The degree of accuracy is controlled by the user through the specification of a maximum allowable error variance. Lossless data compress/reduction is possible to whatever extent the user desires, however, the required quality of the resulting image will obviously be the final deciding factor.

For large data sets (more than 71,000 points) containing lattice data run times varied from 1 to 10 minutes. Further refinements are in progress to automatically account for zonal anisotropic behaviour.

5. References

- Cressie, N. (1991), *Statistics for Spatial Data*. New York, Wiley and Sons.
- Carrera, J., F. Szidarovsky (1985), "Numerical Comparison of Network Design Algorithms for Regionalized Variables," *Applied Mathematics and Computation*, 16:189-202.
- Gonzalez, R., R. Woods (1992), *Digital Image Processing*, Reading, MA, Addison-Wesely.
- Szidarovsky, Ferenc (1983), "Multiobjective Network Design for Regionalized Variables," *International Journal of Mining Engineering*, 1:331-42 (1983)



Figure 2: Original Image and Reductions of 50%, 87.4%,94.3%

Visualizing Cluster Structure in High Dimensions

Evangelos Tabakis
 Department of Mathematics
 Massachusetts Institute of Technology
 Cambridge, MA 02139

Abstract

Assessing cluster structure in dimensions ≥ 3 is difficult. We propose to combine density estimation with the computation of the *minimal spanning tree* (MST) on n iid observations to produce a plot encoding a lot of information in a *two dimensional* curve. The technique involves the recording of changes in the size of the MST edges as the low density observations are *filtered out*. We will discuss:

- The theoretical results that provide the motivation for this approach.
- Interpretation of the plot in various simulated samples.
- Questions of computational complexity.
- Applications to marketing.

1. Introduction

The main problem of cluster analysis is summarized in [MKB79], page 360:

Let x_1, \dots, x_n be measurements of p variables on each of n objects which are believed to be heterogeneous. Then the aim of cluster analysis is to group these objects into g homogeneous classes where g is also unknown (but usually assumed to be much smaller than n).

Several methods have been proposed to tackle this problem. Detailed listings are included in books and review papers such as, e.g., [Eve74], [Har75], [Gor81], [Gor87], [JD88] and [KR90]. Partitional methods, which seek a partition of the sample that optimizes a certain criterion, form one of the oldest groups of methods (e.g. [FR67]). Iterative algorithms must be used to find such optimal partitions (see, e.g., [JD88], page 96 and [KR90], page 102). Alternatively, hierarchical methods use the observations to produce a sequence of n partitions $\mathcal{P}_1, \mathcal{P}_2, \dots, \mathcal{P}_n$ (often referred to as a *hierarchy* of partitions) with the properties:

- \mathcal{P}_1 is the partition into n one-element clusters.

- \mathcal{P}_i has $n - i + 1$ clusters of which $n - i$ are the same as $n - i$ clusters in \mathcal{P}_{i-1} and the $(n - i + 1)$ th cluster is formed by joining the remaining two clusters of \mathcal{P}_{i-1} into one ($i = 2, 3, \dots, n$).

Hierarchical methods have certain advantages that make them popular. Some of them are:

- They describe the *clustering structure* of the data set without the need to *prespecify* the number of clusters we must look for. Choosing the number of clusters can be then based on inspection of the hierarchy of partitions. Note, however, that inspecting the partitions is not a trivial task for large data sets in high dimensions.
- Using the concept of reciprocal neighbors it is possible to form a full hierarchy in $O(n^2)$ steps (see [LMW84], pages 128-129). Partitional methods need iterative algorithms to produce a single partition. Even worse, the work done to compute a partition in, say, three clusters cannot be generally used in calculating a partition in four or two clusters when using a partitional method.
- Identifying clusters is often a subjective decision. What some people may see as one cluster, some others might consider as two or more. It is often a question of how fine a partition we want to find, that determines the answer. This feature of the clustering problem is best captured by hierarchical methods.

In higher dimensions, we can no longer rely on our natural ability to spot cluster structure. Techniques, such as principal components, often implemented to choose interesting projections may also fail to alert us about the possible existence of clusters. In this paper, we are mainly concerned with diagnostic tools which can play this role and guide us in an (interactive) search for cluster structure.

For the reasons mentioned above, we will use a hierarchical method, namely the *single-link* method. It is well known that this method is closely connected to the computation of minimal spanning trees. The approach to be taken is simple: We will rely on the longest edge of

the minimal spanning tree and the way its length varies when we discard observations from low-density regions. We will therefore be using a hybrid method combining density estimation and hierarchical clustering analysis.

2. Minimal spanning trees

The notion of minimal spanning trees formed by points in R^d is quite common but we define it here for completeness.

Definition 2..1 Given a graph $G = (V, E)$ with vertices V and edges E :

- (1) a tree $T = (V_T, E_T)$ is a subgraph of G , (i.e. $V_T \subseteq V$ and $E_T \subseteq E$) which is connected and contains no cycles,
- (2) a spanning tree is a tree for which $V_T = V$,
- (3) a weight function is a function $w : E \rightarrow R^+$,
- (4) the weight of a tree is $w(T) = \sum_{e \in E_T} w(e)$ and
- (5) a minimal spanning tree of (G, w) is a tree T_0 such that $w(T_0) = \min_T w(T)$ where the minimum is taken over all spanning trees of G .

The usefulness of the MST in identifying clusters can best be realized through a couple of examples: In the case of a uniform distribution (which has connected support), the edges of the MST are all comparatively small and about the same size (see Figure 1). In the case of a mixture of two uniforms with disjoint supports, there is one edge of the MST that is quite larger than the others, as can be seen in Figure 2.

How can we make this observation more precise? If a probability measure has a compact support with a finite number of connected components, then it makes sense to talk about the MST computed with vertices these components. Unless the support is connected (one connected component only) the MST has at least one positive edge. If we were to draw a sample from this distribution and compute the MST of the observations we would observe the following:

- The edges of the tree within each connected component are *small* and go to 0 as n goes to ∞ .
- Edges joining observations from different components are *large* and their lengths converge to the distance of the corresponding components.

Let us formulate this into a consistency result:

Theorem 2..1 Let P be a Borel probability measure in R^d whose support is compact and has the connected components $C = \{C_1, C_2, \dots, C_k\}$.

Let $M(P)$ be the length of the longest edge of any minimal spanning tree computed on the set C .

Let $X_1, X_2, \dots, X_n \sim P$ and let $M(P_n)$ be the length of the longest edge of any minimal spanning tree computed

on the set $\{X_1, X_2, \dots, X_n\}$.¹ Then $M(P_n) \rightarrow M(P)$ almost surely as $n \rightarrow \infty$.

Is that, however, the solution to the clustering problem? Unfortunately, Figure 3 shows that things can easily go wrong. Suppose that we contaminate the sample from the mixture of two uniforms in Figure 2 with some additional observations from a bivariate normal. The resulting tree in Figure 3 fails to separate the two clusters by a large edge. In Figure 4 you can compare the box plots from the uncontaminated and the contaminated case. In the contaminated case, no edge stands up as significantly larger than the others!

The problem we run into, is often called the *chaining problem*. A few observations lying between the clusters can form a *chain* through which the MST joins the two clusters without having to use a long edge. It is really a robustness problem.

Theorem 2..2 The gross-error breakdown point of the sequence $M(P_n)$ is 0 for all probability measures P whose support is compact and disconnected.

Proof: For any $\epsilon > 0$, consider, instead of P , the measure $Q_\epsilon = (1-\epsilon)P + \epsilon R_\epsilon$ where $\text{supp}(R_\epsilon)$ is compact, connected and contains $\text{supp}(P)$.

If $X_1, X_2, \dots, X_n \sim Q_\epsilon$ then $M(Q_{\epsilon,n}) \rightarrow M(Q_\epsilon) = 0$ and so $\Pr(M(Q_{\epsilon,n}) > \delta) \rightarrow 0$ for all $\delta > 0$. But $M(P) > 0$, so the breakdown point is 0. \square

3. Dealing with Contamination

The question now is what we can do to cope with this problem. The main idea comes from the observation that *chains* consist of points from *low-density regions*. Were we to use only high-density points to construct our MST, we would avoid chaining! Suppose that P had a density f with respect to Lebesgue measure. Then, the conditional measure $P|f \geq \delta$ for some $\delta > 0$ has density:

$$f_\delta := \frac{f 1_{[f \geq \delta]}}{P(f \geq \delta)}.$$

So, instead of looking only at P (whose support may be just barely connected, just as the support of Q_ϵ in the previous theorem) we can look at $P|f \geq \delta$ for various values of $\delta > 0$. Let $M(P, \delta)$ be the length of the largest edge of the MST computed on the connected components of f_δ . Let $X_1, X_2, \dots, X_n \sim P$ and define $M(P_n, \delta)$ to be the length of the largest edge of the MST computed on the set $\{X_i : f(X_i) \geq \delta\}$.

The next step is to understand the dependence of $M(P_n, \delta)$ on δ . Take the case of a unimodal density f . Then $\text{supp}(f_\delta)$ is going to be connected for all $\delta > 0$

¹ P_n is the empirical measure: $\frac{1}{n} \sum_{i=1}^n \delta_{X_i}$.

and so $\sup_{\delta} M(\mathbf{P}, \delta) = 0$. We would then expect that $\lim_n \sup_{\delta} M(\mathbf{P}_n, \delta) = 0$. But how does $M(\mathbf{P}_n, \delta)$ change with δ ? Here is an answer:

Theorem 3.1 Assume that f_{δ} has a compact and connected support for all $\delta > 0$. If there is a $c > 0$ such that $\forall \delta > 0, \forall x, y \in \text{supp}(f_{\delta}), \exists$ a ball with radius at least $c \|x - y\|$ contained in both $\text{supp}(f_{\delta})$ and in the ball with diameter xy , we have:

$$\Pr \left(\sup_{\delta} \delta M(\mathbf{P}_n, \delta)^d \leq \frac{k}{c^d \omega_d} \frac{\log n}{n} \right) \rightarrow 1$$

as $n \rightarrow \infty$ for all $k > 2$ where ω_d is the volume of the unit ball in \mathbb{R}^d .

What the last theorem states is that $M(\mathbf{P}_n, \delta)^d$ must, eventually, lie below the hyperbola K/δ for a constant K that depends on n but not on δ . Therefore, we know what to expect when plotting $M(\mathbf{P}_n, \delta)^d$ against δ when f is unimodal. We explore this idea in the next section.

4. Diagnostic plots

The conclusion we should keep from the previous section is the following:

- In the case of a unimodal density, the quantity $M(\mathbf{P}_n, \delta)^d$ decreases with δ , roughly like $1/\delta$.
- In the case of a multimodal density, we expect $M(\mathbf{P}_n, \delta)^d$ to increase as the modes become more separated when increasing δ .

In practice, of course, f is not known. Therefore, we will have to use a density estimator f_n instead. Since such an estimator will only be used to decide whether $f(X_i) \geq \delta$, we are hoping that the previous conclusion will not be affected. Suppose we are given $X_1, X_2, \dots, X_n \text{ iid} \sim f$. Let f_n be a uniformly consistent density estimator of f . Let $M(f_n, \delta)$ be the length of the longest edge of the MST computed on the set $\{X_i : f_n(X_i) \geq \delta\}$. Let $\delta_i := f_n(X_i)$, $1 \leq i \leq n$. We choose a positive integer $k \ll n$ and compute $M(f_n, \delta_{(j,s)})$, where $s := \lfloor n/(k+1) \rfloor$ and $1 \leq j \leq k$. Then, we plot: $M_n(f_n, \delta_{(j,s)})^d$ vs $\delta_{(j,s)}$ for $1 \leq j \leq k$ and call this a *cluster plot*.

Before looking at some examples, we have to address the complexity question. Obviously, we can compute one MST at a cost of $O(n^2)$. So a straightforward use of MST algorithms would require $O(kn^2)$ time. However, the following suggestions may, in practice, reduce this cost.

For simplicity, let us assume that $\delta_1 > \delta_2 > \dots > \delta_n$. It is easy to see that if $S_n(f_n, \delta_i) := \{X_j : f_n(X_j) \geq \delta_i\}$ then $M_n(f_n, \delta_i) = d(A_{n,i}, B_{n,i})$ where $d(A_{n,i}, B_{n,i}) :=$

$\max\{d(A, B) : A \cup B = S_n(f_n, \delta_i), A \cap B = \emptyset\}$. So we are led to the following:

ALGORITHM:

- $A_{n,1} := \{X_1\}$, $B_{n,1} := \emptyset$, $M_n(f_n, \delta_1) := 0$.
- For $i = 2, \dots, n$:
 Let $d_{A,i} := \min\{d(X_i, X_j), X_j \in A_{n,i-1}\}$.
 $d_{B,i} := \min\{d(X_i, X_j), X_j \in B_{n,i-1}\}$.
 $d_i := \min\{d_{A,i}, d_{B,i}\}$.
 Assume: $d_i = d_{A,i}$ (the other case is symmetric).
 If $d_i > M_n(f_n, \delta_{i-1})$ then:
 * $A_{n,i} := \{X_i\}$, $B_{n,i} := A_{n,i-1} \cup B_{n,i-1}$.
 * $M_n(f_n, \delta_i) := d_i$.
 - If $d_i = d_{A,i} \leq M_n(f_n, \delta_{i-1}) < d_{B,i}$ then:
 * $A_{n,i} := A_{n,i-1} \cup \{X_i\}$, $B_{n,i} := B_{n,i-1}$.
 * $M_n(f_n, \delta_i) := M_n(f_n, \delta_{i-1})$.
 - If $d_i = d_{A,i} \leq d_{B,i} \leq M_n(f_n, \delta_{i-1})$ then:
 Calculate the MST on $S_n(f_n, \delta_i)$ from scratch to obtain $A_{n,i}$, $B_{n,i}$ and $M_n(f_n, \delta_i)$.

As long as we are adding observations from the same mode, we can expect not to use the third case (which is the only one requiring an $O(n^2)$ step), therefore completing the computations in less than $O(kn^2)$. Let us see how the method works in a couple of examples:

1. First, let us look at a unimodal density, e.g. the bivariate normal. Notice that the support of such a density is not compact, but the support of f_{δ} is compact for every $\delta > 0$. Figure 5 shows the MST for the data while Figure 6 shows the cluster plot ($n = 200$, $k = 30$).

As expected, the cluster plot shows a curve going to 0 roughly like a hyperbola K/δ , attaining the maximum for small δ which corresponds to no truncation. This is a typical **unimodal pattern** easily recognizable.

2. On the other hand, let us go back to the contamination of the mixture of two uniforms. The MST computed on the whole sample didn't help us in identifying the cluster structure (see again Figure 3). However, the cluster plot (Figure 7) is very revealing. The length of the largest edge of the MST is increasing with δ . The maximum is achieved around $\delta = 0.20$ which corresponds to a truncation of 0.8 of the mass. After that we have a sudden drop, a pattern indicating that, at this level, a mode is lost and the conditional distribution $\mathbf{P}[f \geq \delta]$ becomes unimodal with all edges of the MST around 0.

5. Market Segmentation

It is time now to try our diagnostic plots with some real data. Some of the most interesting applications of cluster analysis come from marketing research (see [Chu91] and [PS83]). One particular problem is the use of a questionnaire to obtain information about the existence of segments in a particular market. So, for example, a manufacturer may be interested in identifying homogeneous groups (segments) among clients in order to target them with products particularly suited for those groups. Since the process of developing such *custom-made* products is always costly, the manufacturer must be convinced that such segments do exist. The questions asked in such a research will almost certainly include demographic characteristics (age, income, number of children etc) but other questions may also be included. For an informative introduction to the subject, see [Win78].

In our example, we will use data collected on behalf of Fabhus, an Atlanta manufacturer of prefabricated homes who saw their business decline in the late 80's, after a booming start in the late 70's. The researchers mailed questionnaires to old customers in an effort to reveal the *customer profile* as well as collect information about preferences, previous housing choices and degree of satisfaction. We will concentrate on the demographic questions namely: (1) Age group, (2) number of children, (3) spouse's employment status, (4) profession and (5) income bracket.

Notice that the variables are not continuous but, as we shall see, this does not affect the diagnostic analysis we will perform. As a first attempt to locate clusters (if they exist) we tried to look at scatter plots of pairs of principal components (PC). PC are often successful in identifying directions on which cluster structure is apparent. As you can see, however, in Figure 8, this is not the case here, where little can be said by looking at the first two PC's.

Let us attempt now to construct a cluster plot (Figure 9). The cluster plot shows that the length of the longest edge of the MST begins to increase at about $\delta = 0.01$, reaching a maximum at about $\delta = 0.025$. Since $\delta = 0.01$ already corresponds to a truncation of 0.45 of the mass, we decided to take a look at the data after that truncation (Figure 10). This time the scatter plot of the new PC reveals the existence of two clusters, one substantially larger than the other.

Thus, we have succeeded in visualizing cluster structure in 5 dimensions where the more traditional method (PC) failed. The main use of the diagnostic plots is, as in this last example, interactive. They can guide us in an effort to:

- Decide whether cluster structure exists (by examin-

ing the plot pattern)

- Isolate the structure from contamination.

The important thing to note, is that this can be achieved without being able to actually *look* at the data.

References

- [Chu91] Gilbert A. Churchill, Jr. *Marketing Research, Methodological Foundations*. The Dryden Press, Chicago, fifth edition, 1991.
- [Eve74] B. Everitt. *Cluster Analysis*. Halsted Press, New York, 1974.
- [FR67] H. P. Friedman and J. Rubin. On some invariant criteria for grouping data. *Journal of the American Statistical Association*, 62:1159-1178, 1967.
- [Gor81] A. D. Gordon. *Classification*. Chapman and Hall, London, 1981.
- [Gor87] A. D. Gordon. A review of hierarchical classification. *Journal of the Royal Statistical Society, Series A*, 150, Part 2:119-137, 1987.
- [Har75] J. A. Hartigan. *Clustering Algorithms*. John Wiley, New York, 1975.
- [JD88] A. K. Jain and R. C. Dubes. *Algorithms for clustering data*. Prentice Hall, Englewood Cliffs, NJ, 1988.
- [KR90] L. Kaufman and P. J. Rousseeuw. *Finding Groups in Data*. John Wiley, New York, 1990.
- [LMW84] Ludovic Lebart, Alain Morineau, and Kenneth M. Warwick. *Multivariate Descriptive Statistical Analysis: Correspondence Analysis and Related Techniques for Large Matrices*. John Wiley & Sons, New York, 1984.
- [MKB79] K. V. Mardia, J. T. Kent, and J. M. Bibby. *Multivariate Analysis*. Academic Press, London, 1979.
- [PS83] Girish Punj and David W. Stewart. Cluster analysis in marketing research: Review and suggestions for applications. *Journal of Marketing Research*, 20:134-148, 1983.
- [Win78] Yoram Wind. Issues and advances in segmentation research. *Journal of Marketing Research*, 15:317-337, 1978.

Figure 1: UNIFORM

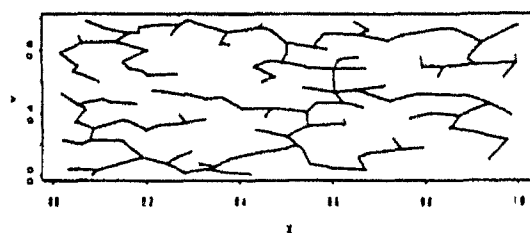


Figure 2: MIXTURE OF UNIFORMS

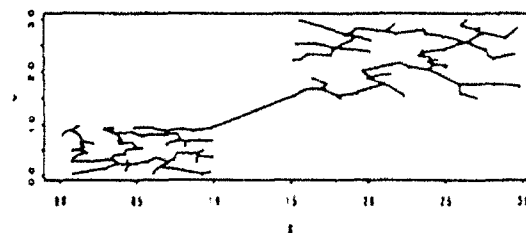


Figure 3: CONTAMINATED MIXTURE OF UNIFORMS

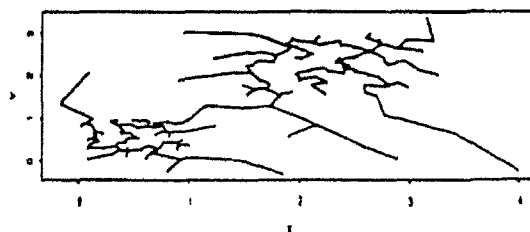


Figure 4: EFFECT OF CONTAMINATION

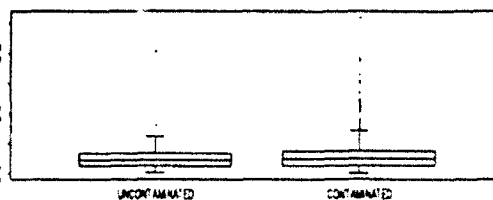


Figure 5: BIVARIATE NORMAL

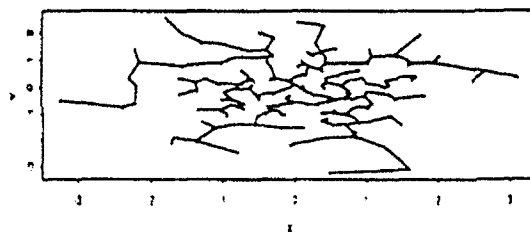


Figure 6: CLUSTER PLOT FOR THE BIVARIATE NORMAL

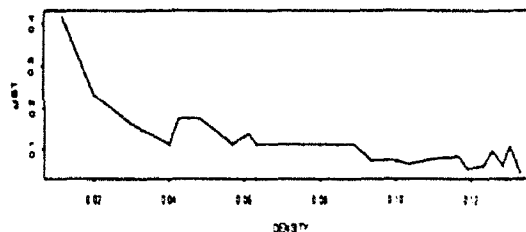


Figure 7: CLUSTER PLOT FOR THE CONT. MIXT. OF UNIFORMS

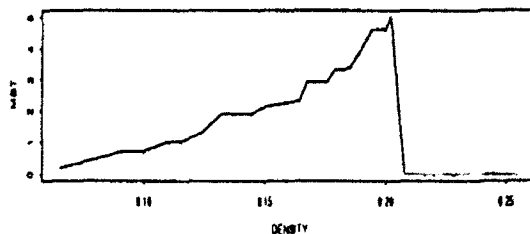


Figure 8: PRINCIPAL COMPONENTS FOR THE FABHUS DATA

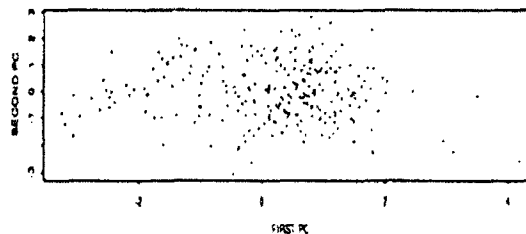


Figure 9: CLUSTER PLOT FOR THE FABHUS DATA

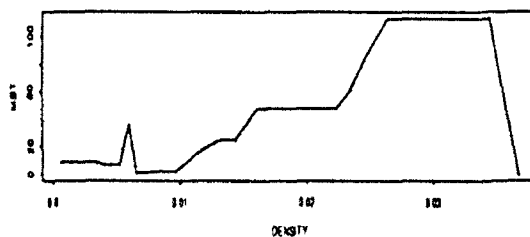
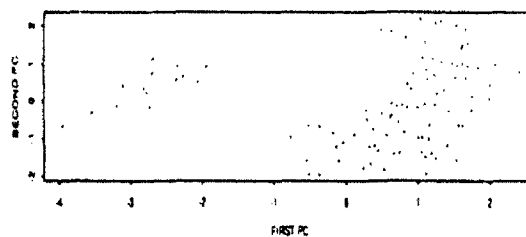


Figure 10: PRINCIPAL COMPONENTS FOR THE TRUNCATED FABHUS DATA



Graphical and Formal Tests of Fit for Discrete Distributions based on Probability Generating Functions

T.W. Epps
University of Virginia
Charlottesville, 22901

Abstract—The adequacy of a probability model for discrete data may be assessed by comparing the empirical probability generating function, $P_n(t)$, with the p.g.f. of the model itself. Both $P_n(t)$ and its model counterpart, $P_0(t; \theta)$, always exist and are continuous on $[0, 1]$. Graphs of $P_n(t)$ and $P_0(t; \hat{\theta})$ against t , where $\hat{\theta}$ is a consistent estimate, afford quick indications of fit. Discrepancies between the functions are easily interpreted, and their statistical significance can be roughly assessed by comparing with approximate confidence limits.

Formal test of fit can be based on various measures of distance between P_n and P_0 . The statistic $I(0, 1) \equiv \sqrt{n} \cdot \int_0^1 [P_n(t) - P_0(t; \hat{\theta})] \cdot dt$, which measures the net area between the functions, is often a simple function of the data, is asymptotically normal, and typically yields tests of much higher power than the traditional omnibus test for discrete data—i.e., Pearson's χ^2 .

1. Introduction

The applied statistician often has to choose an appropriate probability model for a random, univariate sample. Before formally testing a specific null hypothesis, an exploratory graphical analysis may be conducted to see whether the prior belief merits further study. The best-known such graphical method is the probability plot. Applied to location-scale families of distributions, such as

the normal, the goodness of fit can be judged by the near linearity of the plot, and the location and scale can be inferred from the intercept and slope. When dealing with discrete (usually count) data, probability plots are rarely used because the standard models are not location-scale families. Most statisticians just compare the sample and theoretical histograms. This paper proposes using the empirical probability generating function (p.g.f.) as a tool for visual analysis and describes also a formal test based on the p.g.f.

The p.g.f. corresponding to sample x_1, \dots, x_n is $P_n(t) \equiv n^{-1} \cdot \sum t^{x_i}$, $t \in \mathbb{R}$; and that corresponding to model $f(x; \theta)$ is $P(t; \theta) \equiv \sum t^x \cdot f(x; \theta)$. P always exists on $[0, 1]$, and its behavior there fully characterizes f . As the sample mean of i.i.d. random variables, P_n has straightforward properties: $P_n \rightarrow P_0$ uniformly a.s. under $H_0: P = P_0$, and $\sqrt{n} \cdot (P_n - P_0)$ converges weakly on $(0, 1)$ to a mean-zero Gaussian process. Since P_n and P_0 are continuous, their graphs are easily interpreted visually. In many cases a plot of an appropriate transformation of P_n —denoted TP_n —allows parameter estimates to be inferred from the graph, just as from probability plots of continuous distributions. Plots of $P_n(t) - P_0(t; \hat{\theta})$ are also informative and easily interpreted. Just as the chi-squared test formalizes the visual comparison of frequency functions, there is a useful test based on differences in

p.g.f.s. $I(0,1) \equiv \int_0^1 [P_n(t) - P_0(t; \hat{\theta})] \cdot dt$ measures *net* distance between $P_n(t)$ and P_0 on $[0,1]$. It is often a simple function of the data, and its large-sample distributions under H_0 and specific H_1 's are easily found. The test based on $I(0,1)$ is usually much more powerful than the chi-squared test. The next two sections describe the graphical and formal procedures. An application is in Section 4.

2. Graphs of p.g.f.s

Contrasts between Poisson and binomial models will illustrate how plots of TP_n and $P_n - P_0$ can be used to assess goodness of fit. The data are pseudo-random samples of $n=100$ from Poisson (3.) and Binomial (10,.3) laws. The first method depends on finding a transformation T such that $TP_0(t; \theta)$ is a simple function of t and unknown θ . For the Poisson H_0 -- $P_0(t; \theta) = \exp[\theta(t-1)]$ --an obvious choice is $TP_0 \equiv \log(P_0)/(t-1)$. If the model is adequate, a plot of TP_n vs. t should be roughly a horizontal line. Since $TP_n(t) \rightarrow \bar{x}$ as $t \rightarrow 1$, the plot indicates the m.l. estimate of θ . The asymptotic variance of $TP_n(t) - \bar{X}$ is $n^{-1} \cdot \{(t-1)^{-2} \cdot \exp[\theta(t-1)^2] - \theta\}$. Figs. 1 and 2 show plots of TP_n for the Poisson and binomial samples, with pointwise 2- σ confidence limits. The first plot (Poisson sample, $\bar{x} = 2.98$) lies near the horizontal through \bar{x} except near $t=0$. Since $P_n(0) \equiv f_n(0)$, the large value of $TP_n(0) = -\log[f_n(0)]$ for this sample indicates a paucity of zeroes relative to a Poisson with $\theta = 2.98$. One would rightly conclude that the Poisson law describes the data well except at the origin. Fig. 2, on the other hand, shows that it does not adequately represent the binomial data.

A plot of $P_n(t) - P_0(t; \hat{\theta})$ vs. t is also a useful graphical tool. Although requiring θ to be estimated first, the plot is more easily interpreted, and there is no need to find a simplifying transforma-

tion. Figs. 3 and 4 illustrate for the Poisson and binomial data. Dotted lines are 2- σ pointwise confidence limits based on the asymptotic variance, $n^{-1} \cdot \{P_0(t; \theta) - P_0(t; \theta)^2 \cdot [1 + \theta \cdot (t-1)^2]\}$. The Poisson plot lies well within the 2- σ bounds except near $t=0$ --again indicating too few zeroes--while the binomial plot lies everywhere outside. The behavior of $P_n - P_0$ near $t=1$ is revealing also. The k th derivative of the p.g.f. at $t=1$ corresponds to the k th factorial moment. Since for both samples the means of the data and the *fitted* model agree, the slopes of the plots at $t=1$ are zero, and the *curvatures*--second derivatives--indicate differences in dispersion. The linearity of the Poisson plot near $t=1$ shows good agreement between variances, while the concavity of the binomial plot indicates underdispersion.

3. A formal test of fit

Just as the Pearson chi-squared test is based on differences between observed and expected frequencies, the disparities between empirical and theoretical p.g.f.s can be the basis of a formal test of fit. The Kocherlakotas (1986) have devised a quadratic-form test using the p.g.f. Here we briefly describe a test based on the integrated difference. Details are in Epps(1992b).

$I(0,1) \equiv \int_0^1 [P_n(t) - P_0(t; \hat{\theta})] \cdot dt$ measures the *net* difference between p.g.f.s of sample and fitted model on $(0,1)$. This interval is the natural choice, since the p.g.f. necessarily exists there and fully characterizes the distribution. For computation $I(0,1)$ can be expressed as $n^{-1} \cdot \Sigma (X_i + 1)^{-1} - \int_0^1 P_0(t; \hat{\theta}) \cdot dt$. Epps(1992b) gives expressions for the integral term corresponding to several standard distributions--for the Poisson it is $\hat{\theta}^{-1} [1 - \exp(-\hat{\theta})]$. This can always be approximated numerically via the identity $\int P_0(t; \hat{\theta}) \cdot dt = \Sigma (x+1)^{-1} f_0(x; \hat{\theta})$. $\sqrt{n} \cdot I(0,1)$ is asymptotically normal under H_0 if $\hat{\theta}$ is c.a.n. and $f_0(\cdot; \theta)$ is smooth

in θ -conditions that hold when $\hat{\theta}$ is the m.l.e. and f_0 is regular. Since $\hat{\theta}$ is then asymptotically efficient, it follows [Epps(1992a)] that the asymptotic variance, $\sigma^2[I(0,1)]$, equals $\sigma^2[\int_0^1 P_n(t) \cdot dt] - \sigma^2[\int_0^1 P_0(t; \hat{\theta}) \cdot dt]$. The first term on the right can be expressed as $n^{-1} \int \int P_0(st; \theta) \cdot ds \cdot dt - [\int P_0(t; \theta) \cdot dt]^2 = n^{-1} \cdot \{E[(X+1)^{-2}] - E^2[(X+1)^{-1}]\}$. The second term can be approximated to $O_p(n^{-1})$ by expanding $P_0(\cdot; \hat{\theta})$ about θ . For example, in the Poisson case the asymptotic variance of the second term is $n^{-1} \cdot \theta^{-1} \{ \exp(-\theta) - \theta^{-1} [1 - \exp(-\theta)]^2 \}$. H_0 is rejected at level α when $|I(0,1)| / \sigma[I(0,1)]$ exceeds $1 - \Phi(\alpha/2)$.

Epps(1992b) presents Monte Carlo estimates of the test's type-I errors for Poisson, positive Poisson, geometric, and logseries H_0 's. The test is usually very accurate even with samples as small as 50. Power comparisons among many distributions in these families and in binomial, positive-geometric, Neyman Type-A, negative-binomial, Poisson-logseries mix, and zeta families show that the p.g.f. test essentially dominates the chi-squared test. In some cases, as logseries vs. zeta, the chi-squared test is almost useless in small samples, whereas the p.g.f. test has respectable power.

4. An application

The graphical methods and the test are used to assess whether the transaction rate of American Home Products stock follows a Poisson distribution. The number of transactions during 1:00-1:30 p.m. Eastern time was recorded for each of the 243 trading days of 1972. The mean number is 2.235. Fig. 5 shows the difference between the observed relative frequencies and those of a Poisson distribution with parameter 2.235, along with pointwise 2σ confidence limits (open bars). This gives no clear impression that the Poisson model is inadequate, but the plots of TP_n (Fig. 6) and of $P_n - P_0$

(Fig. 7) tell an entirely different story. Their values at $t=0$ show that there is excess frequency at the origin, and the convexity of $P_n - P_0$ near $t=1$ indicates overdispersion. The test statistic based on $I(0,1)$ has the value 3.22, indicating rejection at level .01.

References

- Epps, T.W. (1992a) On a property of min-variance estimators. Manuscript.
- Epps, T.W. (1992b) An omnibus test for discrete dists. using p.g.f.s. Presented at ASA meetings, Boston.
- Kocherlakota, S. and Kocherlakota, K. (1986) Goodness of fit tests for discrete dists. Commun. in Statist., 15, 815-829.

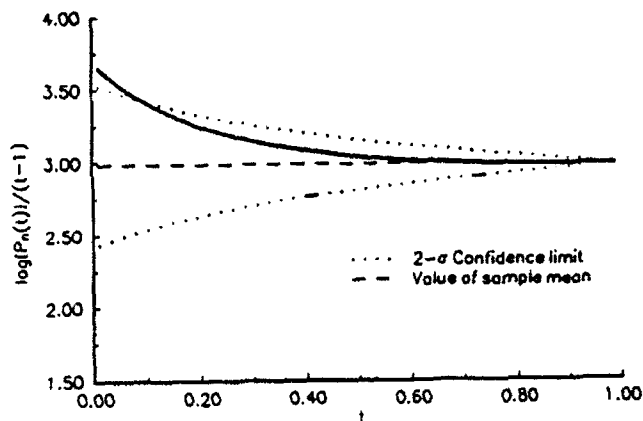
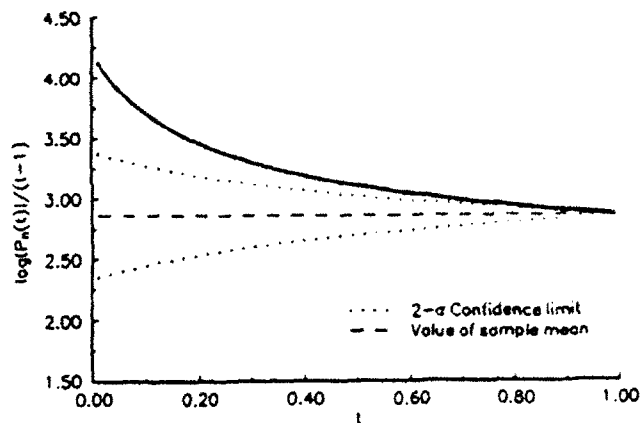
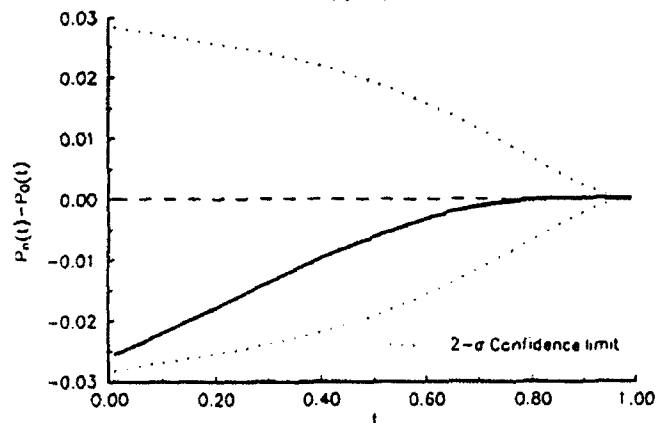
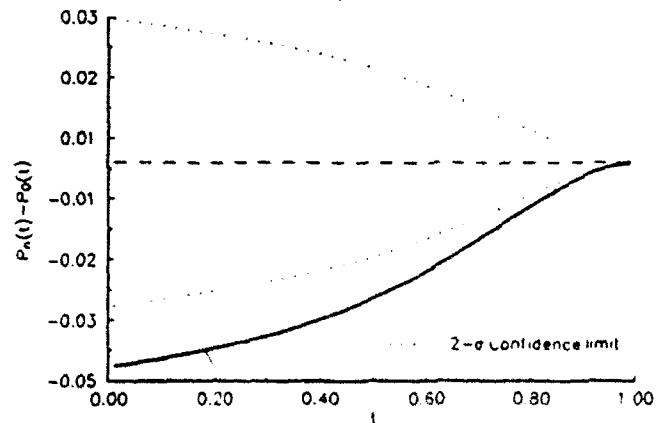
Fig. 1: Transformed p.g.f., $\log(P_n(t))/(t-1)$ Poisson(3) sample, $n=100$ Fig. 2: Transformed p.g.f., $\log(P_n(t))/(t-1)$ Binomial(10, .3) sample, $n=100$ Fig. 3: Difference, $P_n(t) - P_0(t)$, of empirical and Poisson p.g.f.sPoisson(3) sample, $n=100$ Fig. 4: Difference, $P_n(t) - P_0(t)$, of empirical and Poisson p.g.f.sBinomial(10, .3) sample, $n=100$ 

Fig. 5: Difference, $f_n(x) - f_0(x)$, of sample & Poisson relative frequencies

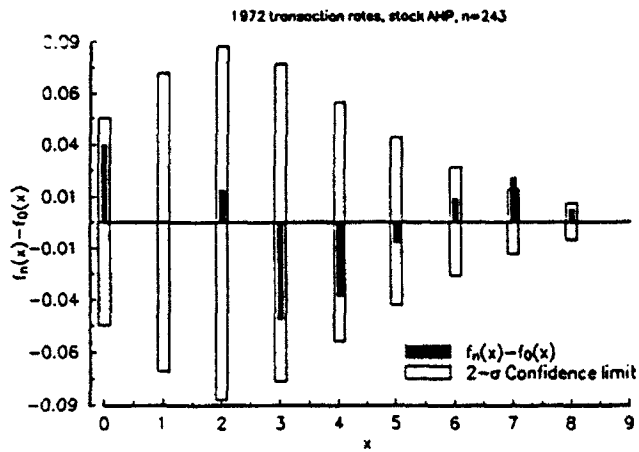


Fig. 6: Poisson-transformed p.g.f., $\log(P_n(t))/(t-1)$

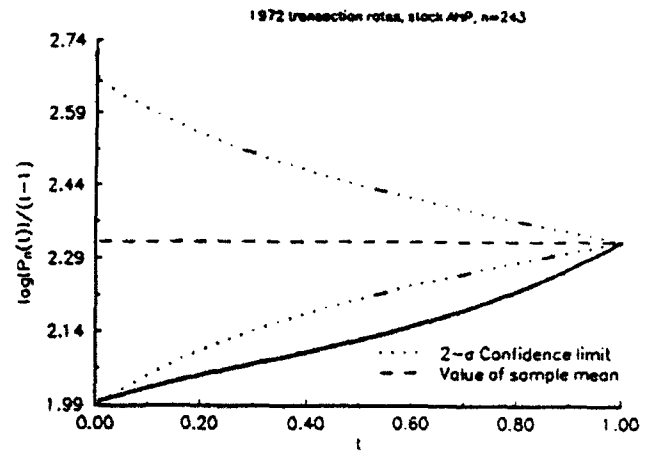
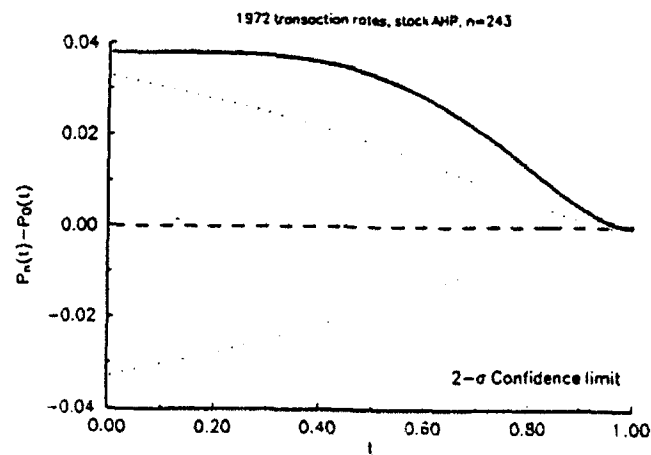


Fig. 7: Difference, $P_n(t) - P_0(t)$, of empirical and Poisson p.g.f.s



Optimal Allocation for Estimating the Product of Two Means

Janis P. Hardwick¹
Statistics Department

University of Michigan, Ann Arbor, MI 48109

Quentin F. Stout²
EECS Department

Abstract

Suppose we wish to estimate the product of the means of two independent populations of Bernoulli random variables, the parameters of which, themselves, are modeled as independent beta random variables. Assume that the total sample size for the experiment is fixed, but that the number of experimental units observed from each population may be random. Using a decision theoretic approach, we seek to minimize the Bayes risk that arises from using a squared error loss function.

Although selecting the form of an optimal estimator is critical to solving this problem, the real difficulty lies in determining an optimal strategy for sampling from the two populations. The problem of optimal estimation reduces, therefore, to a problem of optimal allocation which can be solved exactly using dynamic programming. We utilize similar programming techniques to evaluate exactly some of the other strategies that have been proposed for this problem. (The term *exact* is used repeatedly here to stress the fact that none of the computational results depend on simulation studies.)

1 Introduction

Suppose we have two populations of assembly parts that we intend to use together in the manufacture of a product. Each population has a certain defect rate:

$$1 - p_1 \text{ for Population 1}$$

$$1 - p_2 \text{ for Population 2.}$$

In order for the product to work – both parts must be defect free. We wish to obtain an estimate of the chance that both parts will work by doing some sampling from each population. It is assumed that the sample size for this experiment is a fixed number, N , but that the numbers sampled from each population, n_1 from Population 1 and n_2 from Population 2, may be random.

Now, for Population 1, let

$$X_i = \begin{cases} 0, & \text{if part } i \text{ is defective;} \\ 1, & \text{if part } i \text{ is ok,} \end{cases} \quad i = 1, \dots$$

and for Population 2, let

$$Y_j = \begin{cases} 0, & \text{if part } j \text{ is defective;} \\ 1, & \text{if part } j \text{ is ok,} \end{cases} \quad j = 1, \dots,$$

where the X 's and Y 's are mutually independent. Then at each stage, M , of the experiment, one of the random variables X_M or Y_M will be observed, where

$$(1) \quad \begin{aligned} X_i & \text{ i.i.d. } B(1, p_1) \quad i = 1, \dots \\ Y_i & \text{ i.i.d. } B(1, p_2) \quad i = 1, \dots \end{aligned}$$

Let θ denote the product of the population means
 $\theta = p_1 * p_2$

and define the loss due to any estimate $\tilde{\theta}_N$ by the mean squared error of θ and $\tilde{\theta}_N$

$$L(\theta, \tilde{\theta}_N) = (\theta - \tilde{\theta}_N)^2.$$

We take a Bayesian approach to this problem, assuming that prior information of some sort is available and may be modeled in the form of beta priors on p_1 and p_2 . Let $\xi(p_1, p_2)$, the prior joint density function on $(p_1, p_2) \in \Omega = (0, 1) \times (0, 1)$, denote the product of the independent beta random variables:

$$(2) \quad p_1 \sim \text{Be}(a_0, b_0) \text{ and } p_2 \sim \text{Be}(c_0, d_0).$$

The Bayes risk, \mathcal{R} , for an estimator $\tilde{\theta}_N$ is defined as the expected loss of $(\theta, \tilde{\theta}_N)$

$$\mathcal{R}(\xi, \tilde{\theta}_N) = \mathbf{E}^\xi[L(\theta, \tilde{\theta}_N)] = \mathbf{E}^\xi[(\theta - \tilde{\theta}_N)^2]$$

where the expectation, \mathbf{E}^ξ , is taken with respect to the Bayesian model (2) in which (1) holds conditionally given $(p_1, p_2) \in \Omega$.

2 Minimizing the Risk Function

Minimizing the Bayes risk may be tackled in two stages. The first problem is to find the form of the optimal estimator and the second is to determine how many units to sample from each population. Fortunately, these problems are independent and the first is straightforward.

Suppose that, at any time $M \leq N$, we have observed

$m_1 = \#$ parts from Population 1, and

$m_2 = \#$ parts from Population 2.

¹Research supported in part by National Science Foundation under grants DMS-8914328 and DMS-9157715.

²Research supported in part by National Science Foundation/DARPA under grant CCR-9004727.

Then the *posterior density function*, ξ_M on p_1 and p_2 , is the product of the individual posterior densities

$$(p_1 | i, j) \sim \text{Be}(a, b); \quad (p_2 | k, l) \sim \text{Be}(c, d)$$

where i, j, k and l are sufficient statistics for p_1 and p_2 :

$$i = \sum_{\alpha=1}^{m_1} X_{\alpha}; \quad j = m_1 - i; \quad k = \sum_{\alpha=1}^{m_2} Y_{\alpha}; \quad l = m_2 - k, \\ \text{and} \quad a = i + a_0; \quad b = j + b_0; \quad c = k + c_0; \quad d = l + d_0.$$

Due to the independence of the prior distributions, it is easy to show that, for any $M = 0, \dots, N$, the optimal, or *Bayes*, estimate of θ is simply the product of the posterior means:

$$\hat{\theta}_N = \mathbf{E}^{\xi_M}[p_1] * \mathbf{E}^{\xi_M}[p_2] = \frac{a}{a+b} * \frac{c}{c+d}.$$

What remains to be determined, therefore, is how to carry out sampling from the two populations.

3 Sampling Procedures

A *sampling procedure*, δ , is a sequence of indicator functions, $\delta = (\delta_1, \delta_2, \dots, \delta_N)$, such that

$$\delta_i = \begin{cases} 1, & \text{if Population 1;} \\ 0, & \text{if Population 2.} \end{cases}$$

for $i = 1, \dots, N$ where δ_{i+1} depends only on the information available up until time i . For convenience, we denote partial sequences as

$$\delta(m_1, m_2) = (\delta_1, \delta_2, \dots, \delta_{m_1+m_2})$$

$$\text{where} \quad \sum_{i=1}^{m_1+m_2} \delta_i = m_1 \quad \text{for any} \quad 0 < m_1 + m_2 \leq N.$$

The *Bayes risk* of a procedure δ , denoted by $\mathcal{R}_N(\xi, \delta)$, is the expected loss when δ is the sampling procedure and θ_N the terminal estimate. The problem is to find δ^* such that

$$\mathcal{R}_N(\xi, \delta^*) = \min_{\delta} \mathcal{R}_N(\xi, \delta).$$

One can show that, for any procedure δ ,

$$\begin{aligned} \mathcal{R}_N(\xi, \delta) = & \mathbf{E}^{\xi} \left[\frac{p_1^2 p_2 (1 - p_2)}{(c_0 + d_0 + n_2)} + \frac{p_2^2 p_1 (1 - p_1)}{(a_0 + b_0 + n_1)} \right] \\ (3) \quad & - \mathbf{E}^{\xi} \left[\frac{p_1 (1 - p_1) p_2 (1 - p_2)}{(c_0 + d_0 + n_2)(a_0 + b_0 + n_1)} \right] \end{aligned}$$

where $n_1 = \sum_{i=1}^N \delta_i$ and $n_2 = N - n_1$. However, locating values of n_1 and n_2 to minimize (3) is non-trivial, since the sample sizes themselves are random variables.

A standard approach to finding good or optimal sampling procedures for allocation problems is to locate

asymptotic lower bounds for the Bayes risk of any procedure, and then to seek ad hoc sampling rules that either achieve or come close to achieving the lower bounds. Rekab (1989) takes this approach for estimating the product of two normal means using a quadratic loss function. Hardwick (1990) considers a similar approach in estimating the *difference* of two binomial means using a 'loss plus cost' formulation.

Here, we do not study the asymptotic behavior of sampling procedures, but instead determine *exactly* optimal sequential rules (OS). To establish the potential advantages of using such rules, we define and compare three sub-optimal procedures - Equal Allocation (EA), Best Fixed Allocation (BF), and Local Sequential (LS).

In Section 4, we compute the efficiency of each procedure for various sample sizes and parameter configurations. The *efficiency* of a procedure δ is defined as

$$(4) \quad \frac{\mathcal{R}_N(\xi, \delta)}{\mathcal{R}_N(\xi, \delta^*)}$$

where, for a *fully* efficient sampling procedure, the ratio in (4) is one.

3.1 Best Fixed Allocation Rules

When examining this problem, an obvious question that arises is how well one fares when sampling from each population equally. However, since equal allocation is a special case of fixed allocation (in which n_1 and n_2 must be specified in advance) we consider the more general problem here, and the former in Sections 4 and 5.

For fixed allocation rules, $\delta(n_1, n_2)$, the risk function takes on a simplified form. With n_1 and n_2 no longer random, the Bayes risk reduces to

$$\begin{aligned} \mathcal{R}_N(\xi, \delta(n_1, n_2)) = & \frac{\mathbf{E}^{\xi}[p_1^2 p_2 (1 - p_2)]}{(c_0 + d_0 + n_2)} \\ (5) \quad & + \frac{\mathbf{E}^{\xi}[p_2^2 p_1 (1 - p_1)]}{(a_0 + b_0 + n_1)} - \frac{\mathbf{E}^{\xi}[p_1 (1 - p_1) p_2 (1 - p_2)]}{(c_0 + d_0 + n_2)(a_0 + b_0 + n_1)} \end{aligned}$$

which can be minimized directly since the moments of the prior distributions are constants. In this case, the optimal sample sizes n_1 and n_2 are obtained by solving a quadratic equation.

Note that the third term on the right hand side of (5) is an order of magnitude smaller than either of the first two terms. If we ignore this term and choose n_1 to minimize only the first order approximation of the risk function, the difference in the risks obtained is barely visible (on the order of 10^{-4}). A very close approximation to the best fixed allocation rule, then, is attained if we select n_1 such that

$$\frac{a_0 + b_0 + n_1}{c_0 + d_0 + n_2} = \frac{\sqrt{\mathbf{E}^{\xi}[p_2^2 p_1 (1 - p_1)]}}{\sqrt{\mathbf{E}^{\xi}[p_1^2 p_2 (1 - p_2)]}}$$

3.2 Local Sequential Rules

Because optimal sample sizes actually depend on the true parameter values, we can diminish the risk by using sequential sampling procedures that utilize information in the data as it comes in. Thus, instead of selecting n_1 and n_2 at the beginning of the trial, one can expect to do better by continually re-estimating parameters and basing sampling decisions on the most recent estimates.

One ad hoc approach to sequential sampling is to use a *local* or *myopic* rule (LS). Here, we consider a local rule in which we compare the expected risk associated with each sampling option given the data observed so far. For example, suppose that at time M we have sampled m_1 and m_2 from Populations 1 and 2 respectively; then the *posterior Bayes risk* at M is

$$\begin{aligned} \mathcal{R}_M(\xi_M, \delta(m_1, m_2)) = & \frac{\mathbf{E}^{\xi_M}[p_1^2] \mathbf{E}^{\xi_M}[p_2(1-p_2)]}{(c_0 + d_0 + m_2)} \\ & + \frac{\mathbf{E}^{\xi_M}[p_2^2] \mathbf{E}^{\xi_M}[p_1(1-p_1)]}{(a_0 + b_0 + m_1)} \\ & - \frac{\mathbf{E}^{\xi_M}[p_1(1-p_1)] \mathbf{E}^{\xi_M}[p_2(1-p_2)]}{(c_0 + d_0 + m_2)(a_0 + b_0 + m_1)}. \end{aligned}$$

The expected risk if we sample next from Population 1 and then stop is

$$\mathcal{R}_{M+1}(\xi_M, \delta(m_1 + 1, m_2)),$$

and if we sample next from Population 2 and then stop, we expect

$$\mathcal{R}_{M+1}(\xi_M, \delta(m_1, m_2 + 1)).$$

The myopic rule indicates that we sample from Population 1 if

$$\mathcal{R}_{M+1}(\xi_M, \delta(m_1 + 1, m_2)) < \mathcal{R}_{M+1}(\xi_M, \delta(m_1, m_2 + 1)).$$

The computations necessary to define the LS rule are straightforward, but determining the properties of the rule is more complicated. To obtain the Bayes risk (or other summary characteristics) of the LS procedure, it is necessary to keep track of what can happen at each stage. However, instead of following the sampling process forward in time, as one does in a simulation study, we record what happens beginning from the last stage and work backwards toward the first. This technique is known as *backward induction*; and, while one rarely sees it used to evaluate allocation rules, similar approaches have been taken in Berry and Eick (1987) and Hardwick and Stout (1990).

3.3 Optimal Rules

One expects the LS rule to improve on fixed allocation rules because information about study parameters accumulates and is put to use as the experiment progresses.

To improve still further, however, one needs not only to utilize incoming data, but also to *anticipate* future losses. Suppose, for example, that we are at stage M , having observed (i, j, k, l) . Let $\Delta(i, j, k, l)$ be the set of procedures, $\delta(i, j, k, l) = (\delta \mid i, j, k, l)$, that are consistent with the data at stage M . It is useful to work with the *optimal interim risk* function

$$\mathbf{I}_N^M(i, j, k, l) = \min_{\delta \in \Delta(i, j, k, l)} \mathbf{E}^{\xi_M}[\mathcal{R}_N(\xi, \delta)],$$

which is the minimum expected risk incurred if one were to start at stage M , given (i, j, k, l) , and proceed optimally to stage N . Note that the optimal interim risk at stage N is simply the posterior risk computed for $M = N$:

$$\mathbf{I}_N^N(i, j, k, l) = \mathcal{R}_N(\xi_N, \delta(i, j, k, l)).$$

If we take the next observation from Population 1 and then proceed optimally to the end of the experiment, at stage $M + 1$ we will incur a conditional expected interim risk of

$$\begin{aligned} \mathbf{E}^{\xi_M}[\mathbf{I}_N^{M+1}(i, j, k, l) \mid \delta_{M+1} = 0] = & \\ & \mathbf{E}^{\xi_M}[p_1] \mathbf{I}_N^{M+1}(i + 1, j, k, l) + \\ & \mathbf{E}^{\xi_M}[1 - p_1] \mathbf{I}_N^{M+1}(i, j + 1, k, l). \end{aligned}$$

Similarly the conditional expected interim risk if the next observation is from Population 2 is

$$\begin{aligned} \mathbf{E}^{\xi_M}[\mathbf{I}_N^{M+1}(i, j, k, l) \mid \delta_{M+1} = 1] = & \\ & \mathbf{E}^{\xi_M}[p_2] \mathbf{I}_N^{M+1}(i, j, k + 1, l) + \\ & \mathbf{E}^{\xi_M}[1 - p_2] \mathbf{I}_N^{M+1}(i, j, k, l + 1). \end{aligned}$$

Interim risk obeys the recursive optimality principle

$$\begin{aligned} \mathbf{I}_N^M(i, j, k, l) = & \min\{\mathbf{E}^{\xi_M}[\mathbf{I}_N^{M+1}(i, j, k, l) \mid \delta_{M+1} = 0], \\ (6) \quad & \mathbf{E}^{\xi_M}[\mathbf{I}_N^{M+1}(i, j, k, l) \mid \delta_{M+1} = 1]\}. \end{aligned}$$

This optimality principle is the heart of the dynamic programming approach for determining the optimal rule, working from the end of the experiment towards the beginning. Once the optimal interim risk has been determined for all possible observations at stage $M + 1$, it can then be determined for observations at stage M by using (7). If at stage M we have observed (i, j, k, l) , then the optimal population choice for the next observation is the one which attains the minimum in (7), randomizing in the case of ties.

4 Examples and Heuristics

Often, prior distributions utilized in Bayesian decision problems tend to be chosen more for their mathematical tractability than for their inherent reflection of beliefs

about design parameters. In this problem, however, one finds that a wide variety of perspectives can be represented by judicious selection among the beta priors at our disposal.

Here we examine four different parameter configurations. These examples are intended to illustrate not only how prior configurations may be used to characterize prevailing experience, but also how different prior configurations can affect the sampling schemes discussed in 3.

The first example is relevant when we wish to emphasize differing degrees of faith in our prior knowledge. In such cases, for example, we might have $E^f[p_1] = E^f[p_2]$, but increase proportionately the values of the parameters for the distribution we believe we know more about. The effect of such modeling is that the optimal sampling rule will sample more often from the population with the smaller initial parameters.

Take the case depicted in Table 1, where

$$p_1 \sim \text{Be}(20, 10) \text{ and } p_2 \sim \text{Be}(2, 1).$$

The mean for each distribution is $\frac{2}{3}$, but we have less faith in the accuracy of the prior information for p_2 than for p_1 . While the sampling scheme reflects this, one can see that, as the total sample size increases, the proportions sampled from the two populations approach each other.

Next, consider the case in which $p_1 \sim \text{Be}(1, 2)$ and $p_2 \sim \text{Be}(2, 1)$. In this parameter configuration, one might expect the EA rule to be nearly optimal, since with $a_0 + b_0 = c_0 + d_0$ and $E^f[p_1] = 1 - E^f[p_2]$, the distributions are mirror images of one another. As is evident from the Table 2, however, this is not the case. On average, the optimal rule samples approximately 70% from Population 1, and the EA rule turns out to be only about 85% efficient. This observation leads to a second heuristic regarding sample size selection: if the prior information indicates that one mean is less than the other (all else being equal), an optimal rule will tend to sample more from the population with the lower expected mean.

Our last two examples, Table 3 (in which $a_0 = 0.1, b_0 = 0.01, c_0 = 1, d_0 = 1$) and Table 4 (in which $a_0 = 0.01, b_0 = 0.1, c_0 = 1, d_0 = 1$) may appear at first to be pathological since the small values of a_0 and b_0 indicate that very little information is known in advance. Suppose, however, that the first of the two batches of parts under consideration comes from a manufacturing process that is either *in tolerance* (working) or *out of tolerance* (not working). In such a case, it is the third moment (shape) of the distribution that is of interest rather than the first two moments (center and spread). In the case where $p_1 \sim \text{Be}(0.1, 0.01)$, we face a *u*-shaped

Sample Size	Efficiency (Risk Ratios) (% on Pop 1)			
	OS	LS	BF	EA
$N = 20$	1.000 (10%)	1.000 (10%)	0.927 (00%)	0.792 (50%)
$N = 50$	1.000 (28%)	1.000 (29%)	0.937 (30%)	0.882 (50%)
$N = 100$	1.000 (40%)	1.00 (40%)	0.930 (42%)	0.917 (50%)

Table 1: $p_1 \sim \text{Be}(20, 10)$ and $p_2 \sim \text{Be}(2, 1)$

Sample Size	Efficiency (Risk Ratios) (% on Pop 1)			
	OS	LS	BF	EA
$N = 20$	1.000 (74%)	0.999 (74%)	0.927 (67%)	0.862 (50%)
$N = 50$	1.000 (70%)	1.000 (71%)	0.906 (65%)	0.846 (50%)
$N = 100$	1.000 (70%)	1.000 (70%)	0.898 (64%)	0.837 (50%)

Table 2: $p_1 \sim \text{Be}(1, 2)$ and $p_2 \sim \text{Be}(2, 1)$

distribution where the probability that the process is out of tolerance is ten times greater than the probability that it is in tolerance. After only a couple of observations we should be able to determine, with high probability, which state we have encountered. From that point on, since $p_2 \sim \text{Be}(1, 1)$, we know that we should sample more from the population with the smaller expected mean. Given the circumstances, the fact that the information in the prior with the small parameters is dominated so quickly by the data is precisely what we expect.

5 Conclusions

In writing this paper we were working towards two goals. The first was to solve a fairly straightforward nonlinear estimation problem, and the second was to acquaint the reader with certain useful, but relatively underutilized, computational techniques.

After reducing the statistical decision problem to one of optimal allocation, we considered four classes of rules that could be used to address the problem.

- OS = Optimal sequential rule - defined recursively; located and evaluated via dynamic programming.
- LS = Local sequential rule - an ad hoc myopic strategy which can be located directly (at each stage), but which is evaluated using backward induction.

Sample Size	Efficiency (Risk Ratios) (% on Pop 1)			
	OS	LS	BF	EA
$N = 20$	1.000 (16%)	0.998 (15%)	0.917 (12%)	0.626 (50%)
$N = 50$	1.000 (15%)	0.997 (14%)	0.892 (12%)	0.583 (50%)
$N = 100$	1.000 (14%)	0.997 (13%)	0.878 (12%)	0.565 (50%)

Table 3: $p_1 \sim \text{Be}(0.1, 0.01)$ and $p_2 \sim \text{Be}(1, 1)$

Sample Size	Efficiency (Risk Ratios) (% on Pop 1)			
	OS	LS	BF	EA
$N = 20$	1.000 (92%)	0.997 (92%)	0.823 (33%)	0.754 (50%)
$N = 50$	1.000 (90%)	0.998 (92%)	0.793 (32%)	0.707 (50%)
$N = 100$	1.000 (90%)	0.998 (91%)	0.778 (31%)	0.686 (50%)

Table 4: $p_1 \sim \text{Be}(0.01, 0.1)$ and $p_2 \sim \text{Be}(1, 1)$

- BF = Best fixed allocation rule - defined to be the optimal rule when n_1 and n_2 are fixed in advance. Sample sizes and risk may be obtained directly.
- EA = Equal allocation - defined as any allocation rule such that $\frac{N}{2}$ observations are sampled from each population.

Our statistical conclusions are typified by the results in Tables 1-4. By definition, the OS rule is uniformly best, and subject to heuristic arguments of the sort detailed in previous section, the performance of the other three allocation rules depends on the configuration of the prior parameters and the sample size for the experiment.

In particular, we found that, without exception, the LS rule is so close to being fully efficient that it appears to offer the best overall combination of practicality and efficiency. Furthermore, for a variety of parameter configurations, both the BF and EA rules performed surprisingly well. Still, it is interesting to note how poorly the EA rule performs in settings such as those described in Tables 3 and 4. On the whole, however, given the performance of the LS rule, it seems reasonable to acknowledge that locating the *optimal* rule here is more an academic question than a practical one. Nevertheless, as mentioned, we had another reason for pursuing an entire solution. We believe that this problem serves as an excellent vehicle for illustrating the diverse applications

of the computational technique of backward induction.

For many years, statisticians have known that, in theory, backward induction serves as a useful means for computing solutions to sequential allocation problems. For example, in discrete state space allocation problems, one can often express the optimal solution via a set of recurrence equations that can be solved dynamically. Historically, such programming has been too computationally intensive to be very practical, but increasingly, realistic problems can be solved in this manner. A less well understood ability of backward induction is its capacity to evaluate, *exactly*, properties of all types of allocation rules. Even if an allocation strategy is not defined through a set of recursion equations, one can still determine its attributes through backward induction. This exact approach is highly preferable to the usual method of seeking approximate characteristics of an allocation rule asymptotically or through simulation studies.

References

- [1] Rekab, K (1989), Asymptotic efficiency in sequential designs for estimation. *Sequential Analysis* 8, 269-280.
- [2] Hardwick, J. P. (1989), Computational problems associated with minimizing the risk of a simple clinical trial, *Contemporary Mathematics* 115, 239-256.
- [3] Hardwick, J. P. and Stout, Q. F. (1990), Bandit strategies for ethical sequential allocation. *Symposium on the Interface: Computing Science and Statistics*.
- [4] Berry, D. A. and Eick, S. G. (1987), Decision analysis of randomized clinical trials: comparison with adaptive procedures. *Unpublished manuscript*.

Optimal Adaptive Equal Allocation Rules

Janis P. Hardwick¹
Statistics Department

University of Michigan, Ann Arbor, MI 48109

Quentin F. Stout²
EECS Department

Abstract

Suppose we wish to decide which of two treatments is better, where the outcomes are Bernoulli random variables, the success probabilities of which, themselves, are modeled as independent beta random variables. Assume that the maximal population size for the experiment is fixed, but that the length of the study and the number and order of patients assigned to each treatment may be random. Our goal is to maximize the likelihood of making the correct decision by utilizing a curtailed equal allocation rule, but we wish to do so with a minimal average study length.

We show that this experimental design problem reduces to a problem of optimal adaptive allocation which can be solved exactly using dynamic programming. We compare the optimal allocation procedure to the commonly-used approach of curtailed alternating allocation and show that the optimal allocation procedure is noticeably superior. The evaluations of allocation procedures are all exact, calculated via backward induction. Since the optimal adaptive allocation procedure can be easily determined and evaluated on workstations, and stored on personal computers for ready access during experiments, it is a practical improvement over alternating allocation.

1 Introduction

We are interested in the simple and common decision problem of trying to pick the better of two Bernoulli populations. For concreteness the problem will be described in terms of a clinical trial, but it is equally applicable to other areas such as product testing.

Suppose two treatments are available to treat a certain disease and we need to design an experiment to decide which is better. A maximum of N exchangeable patients can be used in the trial, but the trial may be stopped earlier. (To simplify exposition we assume N is even.) We assume that the patients enter the trial sequentially, and that the outcomes of patients $1, \dots, i$ are known be-

fore patient $i+1$ is assigned. Responses to treatment will be either *successes* or *failures*, where, for each patient, the probability of success on Treatment i is p_i , $i = 1, 2$. Our study design is set up to allow the incorporation of prior information on the success rates p_1 and p_2 . This information is modeled in the form of a joint distribution function ξ on (p_1, p_2) and is taken to be the product of two independent beta random variables:

$$(1) \quad p_1 \sim \text{Be}(a_0, b_0) \quad \text{and} \quad p_2 \sim \text{Be}(c_0, d_0)$$

for $(p_1, p_2) \in \Omega = (0, 1) \times (0, 1)$.

As was mentioned, our main goal is to select the better treatment, but we have a secondary goal as well. We wish to make the decision based on as few patients as possible, and we take advantage of the sequential nature of the data to do this. A *design* is comprised of two parts - an allocation procedure, $\gamma(\xi)$, and a decision rule. An allocation procedure is a rule for deciding what to do at each stage of the trial. Let $\Gamma(\xi)$ represent the class of all allocation procedures with the following features: At any stage i , the procedure may indicate one of three options: assign patient i to Treatment 1, to Treatment 2 or stop the trial. At each stage, the decision of how to proceed may depend only on the prior distribution and the information available from preceding patients.

Regardless of what procedure is used, the form of the optimal decision rule remains the same. The rule states that we select the population with the higher observed mean at the end of the trial.

2 Optimal Decision Making

Unfortunately, our two goals of making good terminal decisions and keeping study size to a minimum are somewhat contradictory, so our concern is to study tradeoffs between these criteria. First, however, it must be noted that, even for fixed sample size experiments, there are no allocation procedures that make optimal decisions for all $(p_1, p_2) \in \Omega$. A useful method of examining optimality in a more restricted sense is to consider procedures that offer optimality along lines of constant difference: $|p_2 - p_1| = \Delta$.

Let $\mathcal{P}^\gamma[(p_1, p_2); N]$ represent the probability of making a correct decision after N observations using procedure γ , when (p_1, p_2) are the true treatment success rates.

¹Research supported in part by National Science Foundation under grants DMS-8914328 and DMS-9157715.

²Research supported in part by National Science Foundation/DARPA under grant CCR-9004727.

Next, define

$$\mathcal{P}_{\Delta}^{\gamma} = \min_{|p_2 - p_1| \geq \Delta} \mathcal{P}^{\gamma}[(p_1, p_2); N];$$

then we say a procedure, γ^* , is Δ -optimal if

$$\mathcal{P}_{\Delta}^{\gamma^*} = \max_{\gamma \in \Gamma} \mathcal{P}_{\Delta}^{\gamma}.$$

3 Curtailed Allocation Procedures

Let \mathcal{C} be the class containing all procedures that are Δ -optimal for all $\Delta \in (0, 1)$. A popular sub-class of \mathcal{C} is the set of *fixed horizon equal allocation rules*, where the term 'horizon' refers to the number of patients actually observed in the study. This class of procedures, \mathcal{C}_{EA} , contains all allocation rules that assign $\frac{N}{2}$ patients to each treatment alternative. Since we are concerned both with making good decisions and keeping study length to a minimum, we work here with a class of procedures that retain the optimal properties of those in \mathcal{C}_{EA} , but which also may be stopped prior to N .

Before characterizing the procedures in this new class, we must first define what is meant by a *state* of the general process that represents our clinical trial. Suppose that we are at Stage M , $0 \leq M \leq N$, of the study and have observed

- i = # patients succeeding on Treatment 1,
- j = # patients failing on Treatment 1,
- k = # patients succeeding on Treatment 2, and
- l = # patients failing on Treatment 2.

Note that $M = i + j + k + l$. The vector (i, j, k, l) which is sufficient for (p_1, p_2) , coupled with the prior density function ξ , is referred to as the state $(i, j, k, l; \xi)$. Thus, at any time M , the state provides all information incorporated in the *posterior density function*, $\xi(p_1, p_2 | i, j, k, l)$, which is simply the product of the individual posterior densities

$$(p_1 | i, j) \sim \text{Be}(a, b), \quad (p_2 | k, l) \sim \text{Be}(c, d),$$

where

$$a = i + a_0 \quad b = j + b_0 \quad c = k + c_0 \quad d = l + d_0.$$

Often, the prior density function will be understood, and only the stage of the trial and the sufficient statistics will represent the state.

If, for some $M \geq 1$, the trial is terminated at a state $(i, j, k, l; \xi)$, then our *decision rule* is to declare as better the treatment with the higher observed mean - selecting

$$\left. \begin{array}{c} 1 \\ 2 \\ \text{tie} \end{array} \right\} \text{ if } \frac{i}{i+j} \left\{ \begin{array}{c} > \\ < \\ = \end{array} \right\} \frac{k}{k+l}.$$

and where we randomize to select a winner if a tie is declared. Throughout, we will assume this decision rule is used whenever the trial is terminated. Note that the state also contains enough information to determine the study length (namely M), the number of failures ($j + l$), and the number of patients assigned to the inferior treatment (either $k + l$ or $i + j$, depending on whether Treatment 1 or 2 is declared the better).

Now, suppose we are using some allocation procedure γ , and that during an experiment a state α is reached. Further, suppose that γ will allocate future patients so that all terminal states that can be reached from α will have the same decision D . In such a case, one can go ahead and make decision D at state α without affecting the Δ -optimal status of γ . This simple concept is called *pruning* or *curtailing*, and we denote the class of all curtailed equal allocation procedures as \mathcal{C}_{CEA} .

Consider, for example, the equal allocation procedure referred to as *alternating allocation*, which is a commonly used oblivious allocation procedure which ignores all previous outcomes and simply alternates back and forth between treatment assignments. If alternating allocation is being used with $N = 10$, and after the 7th patient the state is $(3, 1, 0, 3)$, then no matter what happens on the 8th, 9th and 10th patients (assigned Treatments 2, 1, and 2, respectively), the decision will be that Treatment 1 is the better. Clearly, one may as well prune the decision tree (or curtail the experiment) and declare Treatment 1 the better.

4 Minimum Average Study Length

When considering our second criteria for procedure optimality, study length, we must again resort to a restricted notion of optimality. Since no procedure offers the minimum expected study length for all $(p_1, p_2) \in \Omega$, we turn to a Bayesian concept which we refer to as the *average study length* of a procedure $\gamma: \mathcal{L}^{\gamma}$. Let $L^{\gamma}(p_1, p_2)$ denote the expected number of patients in a trial when the procedure γ is used for the fixed parameter configuration (p_1, p_2) . Then

$$\mathcal{L}^{\gamma} = \int_{\Omega} L^{\gamma}(p_1, p_2) d\xi(p_1, p_2),$$

and it is our goal to find procedures that simultaneously minimize \mathcal{L} and are contained in \mathcal{C}_{CEA} . In other words, we seek procedures $\gamma \in \Gamma^*$, where

$$\Gamma^* = \{\gamma' : \mathcal{L}^{\gamma'} = \min_{\gamma \in \mathcal{C}_{CEA}} \mathcal{L}^{\gamma}\}.$$

Our goal is minimize the average study length among the curtailed equal allocation rules, but one potential difficulty is that there are a very large number of equal allocation procedures, totaling $2^{2^{O(N)}}$. Fortunately the use

Comment: This algorithm determines the minimal average study length, $ML(0,0,0,0)$, among all curtailed equal allocation rules, for a given prior distribution.

For all states α with N patients, $ML(\alpha) := N$.

For $M := N - 1$ down to 0 do

For all states α with M patients

If α *prunable* then $ML(\alpha) := M$
else

If *Treatment 1 permissible*

then $ML_1 :=$ expected minimal study length allocating Treatment 1

else $ML_1 := \infty$.

If *Treatment 2 permissible*

then $ML_2 :=$ expected minimal study length allocating Treatment 2

else $ML_2 := \infty$.

$ML(\alpha) := \min(ML_1, ML_2)$.

Figure 1: Algorithm for Minimal Study Length among Equal Allocation Rules

of sufficient statistics reduces the number of nonequivalent procedures to $2^{\Theta(N^4)}$ since there are only $\Theta(N^4)$ states for which allocation decisions are need.

The algorithm for determining the minimal average study length appears in Figure 1. For the given fixed prior distribution (1), it determines the minimal possible average study length among the class of curtailed equal allocation rules. It works in the following way: For each state α , the algorithm determines $ML(\alpha)$, the minimum average study length given that the study reaches α and then proceeds optimally to the end of the study. The value $ML(\alpha)$ is computed via dynamic programming, which works from the last stage towards the first. The variable M in Figure 1 denotes the number of patients treated so far, and the final value calculated, $ML(0,0,0,0)$, is the answer.

While this is similar to Bellman's classic dynamic programming algorithm, [1], to minimize expected failures, there is a critical difference. With equal allocation procedures, there are states for which allocating one or the other of the treatments is prohibited. To decide if *Treatment 1 permissible*, note that in the state (i, j, k, l) , an equal allocation rule can assign another patient to Treatment 1 if and only if the number of patients that have previously been assigned to Treatment 1 is less than $N/2$; i.e., if and only if

$$i + j < N/2.$$

Similarly, the condition *Treatment 2 permissible* is

$$k + l < N/2.$$

Note that, at any stage $M < N$, at least one treatment must be permissible.

To decide if state (i, j, k, l) is *prunable*, note that one can stop and declare Treatment 1 the better if and only if the number of successes on Treatment 1 exceeds the number of successes Treatment 2 would have if all future assignments to Treatment 2 resulted in successes. That is, Treatment 1 will be the winner in all possible final states that are reachable from this state by equal allocation rules if and only if $i > N/2 - l$. Similarly, Treatment 2 will be the winner if and only if $k > N/2 - j$. While ties can occur, in this problem no state with $M < N$ is *prunable* with a tie outcome; although such states can have tie outcomes among the reachable states, they must also have at least one reachable state with a nontie outcome. Thus the condition α *prunable* is

$$(i > N/2 - l) \text{ or } (k > N/2 - j).$$

Finally, ML_1 and ML_2 are the minimal possible average study lengths if the study reaches (i, j, k, l) and the next patient is assigned to Treatment 1 or Treatment 2, respectively. To determine ML_1 , note that assigning the next patient (patient $M + 1$) to Treatment 1 will either result in the state $(i + 1, j, k, l)$ or the state $(i, j + 1, k, l)$, depending on whether the treatment is a success or failure. Given ξ and M , the posterior probability of success is

$$\frac{a_0 + i}{a_0 + i + b_0 + j}.$$

$N \rightarrow$	20	50	100	200	400
Curt. Alt.	16.2	39.4	78.1	155.3	309.8
Optimal	15.2	36.1	70.8	140.2	278.8

Table 1: $p_1 \sim Be(1, 1)$, $p_2 \sim Be(1, 1)$

$N \rightarrow$	20	50	100	200	400
Curt. Alt.	16.7	41.0	81.4	162.3	324.0
Optimal	15.9	38.4	75.9	150.8	300.5

Table 2: $p_1 \sim Be(1, 1)$, $p_2 \sim Be(25, 25)$

and thus ML_1 is

$$ML_1(i, j, k, l) = \frac{a_0 + i}{a_0 + i + b_0 + j} \cdot ML(i + 1, j, k, l) + \frac{b_0 + j}{a_0 + i + b_0 + j} \cdot ML(i, j + 1, k, l).$$

Similarly, ML_2 is given by

$$ML_2(i, j, k, l) = \frac{c_0 + k}{c_0 + k + d_0 + l} \cdot ML(i, j, k + 1, l) + \frac{d_0 + l}{c_0 + k + d_0 + l} \cdot ML(i, j, k, l + 1).$$

The assignment $ML(\alpha) := \min(ML_1, ML_2)$ is the essence of the *principle of optimality* which is being exploited by dynamic programming. The principle indicates that the minimal average study length of any study passing through the given state is the smallest of the minimal average study lengths for each permissible treatment.

The algorithm does not explicitly show the allocation procedure that achieves the minimal average study length, but it can be determined by noting for each state whether the algorithm decides to terminate or, if not, which treatment gives the smaller expected study length. This information can be stored and used to conduct the study.

5 Results

In Tables 1-4 we have given results comparing curtailed alternating allocation to the optimal curtailed equal allocation. Each table corresponds to a different configuration of the prior parameters a_0, b_0, c_0, d_0 , and gives the average study length for several different values of N . All the entries are exact to within roundoff errors.

The basic heuristics to be learned from the tables are as follows:

1. For similar means, smaller variances correspond to greater average study lengths;

$N \rightarrow$	20	50	100	200	400
Curt. Alt.	16.1	39.2	77.7	154.6	308.3
Optimal	15.1	36.1	71.0	140.7	280.0

Table 3: $p_1 \sim Be(1, 1)$, $p_2 \sim Be(40, 10)$

$N \rightarrow$	20	50	100	200	400
Curt. Alt.	18.0	44.6	88.9	177.5	354.7
Optimal	16.1	38.7	76.3	151.6	302.2

Table 4: $p_1 \sim Be(4, 1)$, $p_2 \sim Be(40, 10)$

2. As the difference between the means becomes smaller, the average study length becomes greater.

The first point is illustrated by comparing Table 1, where both treatments have a uniform prior, with Table 2, where Treatment 2 has a prior whose mean is still $1/2$ but whose variance is much smaller. The second point is illustrated by comparing Tables 2 and 3, where in Table 3, Treatment 1 continues to have a uniform prior, but now Treatment 2 has a prior with a mean of $4/5$.

Finally, in Table 4, we are trying to choose the better treatment when one has a prior of $Be(4, 1)$, and the other has a prior of $Be(40, 10)$. Comparing to Table 3, we see that the equal means in Table 4 increases the expected study length for both procedures, and does so much more dramatically for alternating allocation than for the optimal allocation.

6 Further Remarks

In this short conference paper, we have illustrated only the ability to minimize the average study length within the class of curtailed equal allocation rules, which were selected because they are Δ -optimal. However, the same algorithmic approach can be applied to other situations. Within the class of curtailed equal allocation rules, one can apply the constrained dynamic programming method described in Section 4 to such problems as

1. Minimizing the expected number of failures;
2. Minimizing the expected number of patients assigned to the inferior treatment; and
3. Minimizing the expected cost of the experiment when different costs may be incurred for each different (treatment, result) possibility,

where expectation is taken to mean average with respect to the prior distribution on (p_1, p_2) .

For these alternative optimizations, the only changes needed are in the assignment of ML at states that are

terminal. For example, to minimize the number of failures, the new value of ML at terminal states is just $j + 1$.

Another extension that is easy to incorporate is that in which a third decision is allowed at the end of the study. For example, rather than randomly picking one of the treatments as better if the observed means are similar, one may prefer to declare a study outcome of "no difference". In this case, a new decision rule would be specified and the test for prunability of a state would become slightly more complicated. Otherwise the algorithm would be just as in Figure 1.

There are many other easy variations. For example, the present program allows us to optimize on one criteria, such as study length, and then to evaluate the resulting procedure on other criteria, such as number of failures. Another goal may be to determine how robust a design is. Here, one can create a procedure using one set of priors but evaluate it using a different set. All such evaluations would be exact, performed using backward induction.

Historically, the use of backward induction to help solve sequential allocation problems has been limited due to the method's computational intensity. Lately, however, it has become quite practical given the current speeds and memory capacity of computers. For example, on a large workstation one can carry out designs and evaluations for $N > 400$, and on a parallel computer one can handle $N > 1000$. Further, having used a workstation or departmental computer to design and evaluate an allocation procedure for, say $N = 400$, the procedure can be compressed and stored on a personal computer, available for ready access in many testing situations. These computational points will be explored more fully in an expanded version of this paper and in [3].

The binomial selection problem has been examined in settings far too numerous to note here. One paper that is particularly worthwhile to reference here, however, is an unpublished work of Berry and Eick [2] that was called to our attention during the presentation of the present paper. The special relevance of Berry and Eick [2] is that it is one of the few papers of which we are aware that utilizes backward induction to perform exact evaluations of non-recursive procedures. In their paper, the goal was to attain a tradeoff between failures and the probability of selecting the better treatment, without curtailment. Instead of maintaining optimal probability of selecting the better treatment by searching through equal allocation rules, dynamic programming was used to minimize the sum of the number of failures observed during a study of N patients, plus the expected number of failures in a future population of size P , assuming that the treatment declared better from the study was used for this population. The expected failure rate in the second population

was the posterior estimate of the failure rate for the better treatment, based on the prior and the observations during the study.

Intuitively, in such an approach, for fixed N , increasing P increases the importance of correctly selecting the better treatment, while decreasing P increases the importance of minimizing failures during the initial study. While this idea is certainly not unique to Berry and Eick [2], the approach that they take with respect to "selecting the better treatment" is a Bayesian notion. In this paper we use a frequentist version of the properties of the decision rule and a Bayesian version of average study length. This latter approach is part of a general program of blending Bayesian and frequentist ideas in both the design and analysis phases of an experiment. Such a blending, while often controversial, is highly flexible. Not only can it be useful in a wide range of applications, but it can also yield designs satisfying a variety of statistical criteria.

References

- [1] Bellman, R. (1957) *Dynamic Programming*. Princeton University Press.
- [2] Berry, D.A. and Eick, S.G. (1987), Decision analysis of randomized clinical trials: comparison with adaptive procedures. *Unpublished manuscript*.
- [3] Hardwick, J. and Stout, Q. F. (1992), Computational aspects of sequential allocation for testing with multiple criteria. In preparation.

History of Statistics in Real Time: Hammers and Nails

After Dinner Talk

by

Emanuel Parzen

Department of Statistics

Texas A&M University, College Station, TX 77843

Contents

1. Is what statisticians do fun?
2. Statistical hammers and nails can help guide what statisticians do.
3. Hammers that make statisticians dull.
4. History of Statistics in Real Time and Stigler's Book
5. Esoteric research should provide exoteric hammers.
6. Quality control of the art of statistical science.

1. Is what statisticians do fun?

What statisticians do is even analyzed in comic strips! I quote "Miss Peach: Arthur's Career Advice".

Question: "Arthur, I'd like to become a statistician. What are my chances?"

Answer: "If you don't know, you've made the wrong choice."

A statistical tradition is that a statistical talk should begin with a joke. I can tell many jokes of the form: "How many statisticians does it take to change a light bulb?" One answer: "Can we get back to you on that; our computers are working it out."

But instead of telling you a cryptic joke, I would like to begin my talk just as cryptically by telling you the proverb which motivates my title.

"To the statistician with a hammer, every problem is a nail."

My talk will propose a revised proverb:

"The statistician needs a unified set of classical and modern hammers, so that almost every problem can be transformed to a nail. The statistician needs nails, so that the hammers can be fun (functional and fundamental). The computational statistician needs graphs which are fun in the sense that they represent plots of functions."

Let me note that the Box Plot introduced by Tukey may not be fun (a representation of a function). I propose a Density Quantile Box Plot.

An after dinner talk traditionally should make people laugh. After all, it should be a fun talk.

Short run fun is provided by the stand up comic approach about statistical personalities. (I could tell you a story about a Fellow Texan or a Fellow Texan).

Among performers who make people laugh one can distinguish comedians, comics, and humorists. A comedian says *funny things*. A comic says *things funny*. A humorist *delivers a message*. All tailor their material to fit their audiences. Let me adapt a story told by humorists.

In my story the cast is computational statisticians from Texas, the San Francisco Bay Area, New Jersey, and New England who met tonight before dinner at the drinking hour.

My life has been interesting because I have lived on all four American coasts (East, West, Great Lakes, and Gulf), have Mathematics degrees from Harvard and Berkeley, been a Statistics Department faculty member for many years at a diversity of universities (Columbia, Stanford, SUNY Buffalo, and Texas A&M), and visited widely (geographically and intellectually).

The Texas computational statistician poses as a fisherman. He says to the West Coast statistician: "You should have been with me last night. I caught a 65 pound catfish." They are joined by the New Jersey statistician, and the Texan tells him: "I caught a 35 pound catfish last night." When the statistician from New England joins them, the Texan tells him: "You should have been fishing with me last night. I caught a 25 pound cat." The Californian exclaims: "That fish of yours is kind of shrinking, isn't it." The Texan explains: "I learned a long time ago that you never tell a man more than you think he'll believe."

As an example of how art stimulates life, let me note how I am stimulated by the reverse word order definitions of a comedian (funny things) and comic (things funny) to conjecture reverse word order definitions of statistical computation (computation of statistics) and computational statistics (statistics from computation). "Statistical computation" uses computing science to conveniently compute statistics, while "computational

statistics" creates new (often, computer intensive) statistics made possible by advances in computers, computer science and signal processing.

2. Statistical hammers and nails can help guide what statisticians do.

My approach in my after dinner talk is to seek long run fun by preaching "fun statistics". My goal is to offer you the equivalent of a *fortune cookie* which clearly is appropriate as after dinner fun.

The good fortune that I would like to offer you tonight is one that I believe statisticians would like to have at their side as they take the many "Oral Exams" that a statistician encounters in a long career. These exams occur during the statistician's interactions with clients with whom one is collaborating or with whom one is counseling on how to apply statistical methods.

Your fortune cookie of the night reads:

"You have good friends who will come to your aid in time of need."

Tonight I would like to introduce a good friend which may be able to provide guidance to the past and future (real time) practice of statistics. This friend is your appreciation of the history of statistics and classifying what happened in the past (and what ought to happen in the future) as being either new hammers or new nails.

Hammers are instruments or tools or "statistical data-scopes" analogous to telescopes and microscopes. *Nails* are the applied phenomena and data that we research with these instruments, tools, or scopes.

When one proposes a symposium to a funding agency, they demand a justification; "what's new to discuss?" they ask. The answer can be either "new theory and methods (new hammers)" or "new applications and data (new nails)".

Early in my career in my interaction with electrical engineers I would try to clarify our relationship: "Why is it if I know something, you call it 'theory' while if you know it you call it 'applied'?" I would explain this situation today by saying that something known to the statistician is usually a hammer, and something known to the applied researcher is usually a nail.

It should be noted that there are several levels of being a hammer (theoretical) and a nail (application). There is an exciting new field of approximation theory called "wavelet theory". It is a mathematical hammer developed by applied mathematical scientists to represent and approximate a function defined abstractly. It is a mathematical nail and simultaneously a statistical hammer when used by statisticians; my personal research goal is to develop nails which are statistically meaningful functions (such as the comparison density or change density)

to estimate (often using wavelet or kernel estimators). Another statistical nail (for the hammer of wavelet estimation) is dependence density functions that disclose or expose the regions where two variables X and Y have a positive association.

In 1955 I was starting to do research on the foundations of modern time series analysis (spectral analysis, extraction of signals from noise). To my statistical colleagues I said in jest "After all, regression analysis is the discrete parameter analogue of continuous parameter time series analysis". I strongly recommend that in 1992 this jest should be taken seriously; as we study wavelet theory (and reproducing kernel Hilbert spaces) we should ask what are the analogues in regression analysis.

3. Hammers that make statisticians dull.

One problem that statisticians traditionally seem to have is that they may have overdeveloped the art of insulting each other's work. Henry Mann (a mathematician who spent his mid-career in statistics) said to me in 1967 after a talk I had just given:

"That's why I left statistics. Instead of discussing what you did, they discussed why you should not have done it."

If one does theory, the reaction one usually gets to one's work is a three pronged attack:

- "It won't work."
- "If it works, it's not new."
- "If it's new, it's trivial."

If one does applied work, typical of our colleagues' attacks is the attitude expressed in the proverb that I quoted at the beginning of my talk:

"To the statistician with a hammer, every problem is a nail."

One interpretation of this criticism is one often expressed by dissatisfied clients of a statistician (denoted S). " S hardly listened to my problem (nail), before assuring me he had the solution (hammer) already available in his tool kit." The motto of a statistician who works as a consultant is equivalent to: "have hammer, will travel."

This criticism compares statisticians to Procrustes, the mythical inn keeper, who had one bed in his inn which he transformed all travelers to fit; his motto was "one bed fits all".

I mention the Procrustes criticism because I believe it illustrates the main point of my talk which is that to understand the many influences that have contributed to the past history of modern statistics, and can guide

the development of its future history, we should think of new developments as being of two types which we can call hammers and nails. If we think this way then we might conclude that graduate education in statistics is dull because it has too much recipe theory and recipe methods!

I believe that computational statisticians are more ready to implement education in "adaptive theory" which integrates theory and methods and makes it easier to adapt theory to new problems and thus have the ability (on the occasions when it is needed) to provide each nail (client) with custom made hammers (rather than off the shelf hammers) adapted for realistic probability models for the client's nails.

4. History of Statistics in Real Time and Stigler's Book

The history of statistics is being increasingly fashionable to study. Important and exciting books are

Todhunter. (1865). *A History of the Mathematical Theory of Probability from the Time of Pascal to that of Laplace*.

David, F. N. (1962). *Games, Gods and Gambling*.

Hacking, Ian (1975). *The Emergence of Probability*

Stigler, Stephen (1986). *The History of Statistics: the Measurement of Uncertainty before 1900*.

Porter, T. M. (1986). *The Rise of Statistical Thinking: 1820-1900*.

Edwards, A. W. F. (1987). *Pascal's Arithmetical Triangle*.

Daston, Lorraine (1988). *Classical Probability in the Enlightenment*.

Hald, A. (1990). *A History of Probability and Statistics and Their Applications before 1750*.

About the history (in real time) of computational statistics we are fortunate to have available two important and marvelous books:

Tufte, Edward (1983) *The Visual Display of Quantitative Information*

Tufte, Edward (1990) *Envisioning Information*.

What I would like you to remember from my talk is my concept of "history of statistics in real time" which proposes that we should study history from the point of view of how it guides us in our current practice of the discipline and profession of statistics, and to accomplish this the history of statistics should be appreciated as the history of hammers and nails.

I believe that evidence for my proposition can be provided by quoting the last 3 pages (pp. 358-361) of Steve Stigler's impressive 1986 book (*The History of Statistics: The Measurement of Uncertainty before 1900*). My interpretation of Stigler's summary: during the period 1750-1900 in which statistics developed into a science, there was one hammer (it was least squares all the way) and a progression of nails (the fields of application of the developing science of statistics changed over many fields from astronomy to social sciences to genetics).

Yule's work provides a symmetric ending to this narrative: We began this history with the method of least squares and we shall end with the method of least squares. Yet the story is far from being circular. Over a two-century period there had been sporadic progress in the measurement of uncertainty, a sequence of developments we may think of as leading toward a completion of the logic of the quantification of science, by eventually permitting the formal evaluation and comparison of measurements. In this sequence the early achievements in astronomy and geodesy were in some respects like the later successes in the social sciences. In both cases a key barrier had been the lack of a conceptual structure that permitted the combination of observations; and in both cases the theory of probability had played a crucial role in overcoming the barrier, with the method of least squares supplying the means for completing the calculations. Beyond these broad similarities, however, lay a number of important differences.

In astronomy the combination of observations had required both the anchor of the mathematical theory of gravitation and the growing knowledge of the behavior of random sums. The theoretical structures of celestial mechanics not only defined the goals for the astronomers' empirical work but also helped the astronomers to reach those goals. By giving a link between different measurements of the same body, Newtonian theory provided a route by which the measurements could be combined, a way in which the relatively small numbers of major causes could be incorporated in one equation and related to the observations. Yet, as the example of Euler shows, that link was not enough by itself. A theory of errors was also needed, both the idea that combining measurements was beneficial, not harmful, and a means for turning the combination to inferential use.

Mayer grasped this intuitively in 1750 and a few years later Simpson added a formal analysis for simple means, but it took over a half-century before the grand Gauss-Laplace synthesis of 1810 was achieved. The delay is ample testimony to the difficulties of formulating, much less solving, the problem. Nevertheless, by the time of Laplace's death in 1827 a major success had been recorded, and the use of probability to measure, compare, and interpret uncertainty was well on the way to becoming a commonplace in astronomy and geodesy.

In the social sciences the problem took on a different face. It was not that theory was lacking: By the middle of the nineteenth century several economists had given mathematical expression to the theories of Adam Smith and David Ricardo. But even though these theories might have captured the major causes that were of interest to the economist, they did not incorporate the myriad causes of little concern in economic theory that nonetheless had a major impact upon the data used to test that theory. Cournot could relate price and demand in theory, all other things being held constant, but he could not hold all other things constant in the real world. A new conceptual approach was needed, and it arose from an unexpected quarter—from studies of heredity.

Galton, Edgeworth, and Pearson assembled the structure; Yule completed it by finding a variation on their advances that finally provided a formulation and analysis for questions in the social sciences. It is ironic that in some respects the tool Yule used was an old tool, the method of least squares. Of course, the matter was much subtler than that; it was not Legendre's or even Gauss's or Laplace's least squares that Yule found, but a superficially similar tool that had transformed by the concepts developed by Galton and others. Legendre's least squares had been around and freely available for ninety-five years. But when it had been tried before, in isolated instances, it had not answered the right questions. By 1900, though, the questions could be reformulated so that the astronomer's least squares could be used to new purpose. It was not merely an elegant variation of language that called for the term *regression analysis* for this conceptually new use of least squares. In 1805 the coefficients were constants, deriving their mean-

ing from an exterior theory. In 1900 the coefficients were interior theoretical constructs, given their meaning in the context of Galton's variance component models as what we now characterize as conditional expectations of multivariate distributions. In this new framework least squares provided the homogeneous sub-classifications for analysis Quetelet had sought while offering a sufficiently restricted setting that Cournot's worries about post-data selection could be addressed in at least limited ways.

The realization that the regression and correlation concepts that had emerged in Galton's studies of heredity were intimately related to the least squares of the beginning of that century was the second great synthesis of the history of statistics. In this second synthesis, probability had played a role rather different from that in the theory of errors. In the theory of errors, the probability calculus had revealed order in chaos with the central limit theorem, and that discovery had made the measurement of uncertainty in aggregate measurements possible. In the social sciences the same magnificent theorem had posed a problem of seemingly incredible difficulty: How could the known diversity of causes be reconciled with this always present order? How could the normal distribution Quetelet had found be disassembled to allow a study of causes? Galton's quincunx had led to the answers to these questions, by suggesting a new role for conditional probability. In the theory of errors, conditional probability had permitted inference about the constants of astronomers' theories. In regression analysis, conditional probability made possible the very definition of the quantities about which the statistician was interested in making inferences.

Looking back, we can see four distinctly different solutions to the problem of the combination of observations, four ways in which the variation in external factors could be allowed for in order to permit an attempt at aggregation: through external theory (as in astronomy); through experimental design (as in psychology); with internal regression analysis; and with large amounts of multiply classified data, through fine cross-classification. Actually, the last of these was not really feasible in the nineteenth century. When data were plentiful (for example, a census), it was in principle possi-

ble to look at all manner of categories; but in practice it was not. In regard to the U. S. census, Herman Hollerith wrote, "Until the census of 1890, we never even knew the proportion of our population that was single, married, and widowed . . . To have divided the native born into those of native parentage and those of foreign parentage, would have been practically impossible with the methods of 1880. To obtain the population classified according to age, sex, and birthplace of mother could not have been considered" (Hollerith, 1894, p. 678). This direct assault upon the de Keverberg dilemma would require modern tabulating equipment, even when the data were plentiful and at hand.

The conceptual triumphs of the nineteenth century had been the product of many minds working on many problems in many fields, and one of the most striking of their accomplishments was the creation of a new discipline. Before 1900 we see many scientists of different fields developing and using techniques we now recognize as belonging to modern statistics. After 1900 we begin to see identifiable statisticians developing such techniques into a unified logic of empirical science that goes far beyond its component parts. There was no sharp moment of birth; but with Pearson and Yule and the growing numbers of students in Pearson's laboratory, the infant discipline may be said to have arrived. And that infant was to find no shortage of challenges.

5. Esoteric research should provide exoteric hammers

My current research program, which I call Change Analysis in the wide sense, has among its goals the unification of parametric and nonparametric statistical methods for discrete and continuous data and the detection and measurement of change. Our basic goal is a framework which emphasizes the development of "analogies between analogies" (a concept created by von Neumann and Ulam to describe the next level of mathematical theory after "analogies between theorems").

I am excited by the "analogies between hammers" that can be developed between four fundamental distinct theories of statistical function smoothing:

1. Probability density estimation
2. Spectral density estimation
3. Nonparametric regression (fixed effect)

4. Nonparametric regression (random effect).

I would like to make a plea that (before the year 2000) the fundamental statistical hammers of function estimation should become *exoteric* (consumer products for applied statisticians) rather than *esoteric* (just an intellectual game of specialized smoothing statisticians). Even members of the esoteric group may not be aware of the latest recommendations for algorithms to compute kernel estimators.

The concepts of esoteric research yielding exoteric hammers (spinoffs) are intended to raise our consciousness about a research trend that often becomes apparent to senior scholars with many years experience: *the researchers in a field individually prosper doing research which adds to their reputation but the field as a whole is not prospering in the sense that it is not doing important things that have an impact outside of the field.* I believe that there is a remedy: researchers in the field should organize to do strategic planning, to act collectively to define the missions of the field and to ensure that things get done that (under the system of individual initiative) are alleged not to add to one's reputation.

Examples of how groups can aim to continuously improve the quality of their field: (1) stimulate expository review articles which call attention to methods which are the "best" hammers for applications and to important advice that were once published somewhere (and often are not given exoteric exposure because they are part of the folklore known to esoteric researchers); (2) raise consciousness about good taste in choice of research problems (which research problems are internally generated rather than externally generated, and which optimization criteria have scientific significance).

Modern British poets complained that modern mathematicians were better treated because clerks would write letters to newspapers complaining that they did not understand modern poetry but never complained about not understanding modern mathematics. The future of statistics (and of computational statistics) requires the statistical public be sufficiently concerned to write letters complaining that (1) esoteric statisticians should be more concerned to be exoteric by constructing statistical hammers more applicable to increasingly complex statistical nails, (2) applied statisticians should be more statistically cultured and more effectively use the continually improving statistical hammers available, and (3) computational statistical software packages should help achieve goals (1) and (2).

I believe that to succeed one has always needed (in addition to talent) a "hook" which promises an impact. Being an eternal optimist, I believe that the future is very promising for those mathematical, statistical, com-

putational scientists who are aware of both esoteric and exoteric goals.

6. Quality control of the art of statistical science

Question: Why do I propose that hammers and nails are useful concepts to describe respectively methods and applications in the discipline and profession of statistics?

Answer: They can aid statisticians' understanding (and public understanding) of past and future history of statistics, can help us communicate with the public, and can help provide control procedures to continuously improve the quality of the art of statistical science. The means by which statisticians can achieve these ends is to practice fun statistics that is elegant (award winning hammers) as well as useful (award winning nails).

Statisticians need to support each other with more awards. To improve their image (and avoid decline) statisticians must improve how their contributions are recognized and propagated by other researchers.

The future of the statistical sciences requires: improvements in hammers; improvement in abilities to transform problems to nails; and improvements in statistical history, culture, and communication. I hope I have stimulated your interest in discussing with your colleagues the question of how to improve the quality of the arts of statistical science and computational statistics.

In this talk on the history of statistics in real time it is very appropriate to pay tribute to the vision and initiative of the pioneers of the annual Interface symposia (begun in 1967 in Southern California by Arnold Goodman and Nancy Mann) and to those who created the Interface Foundation to perpetuate the vision and initiative. The tradition of great organizers has been maintained this year by Joe Newton (who is also my esteemed colleague and friend, Head of the Statistics Department at Texas A&M, and author of the important book *TIMESLAB*). Let's toast Joe!

I would like to conclude with a computational statistics variation of the statistical joke with which I began this talk.

One statistician to another: Colleague, I'd like to become a computational statistician; what are my chances?

Answer: If you don't know, you should attend the next Symposium on the Interface of Computer Science and Statistics.

INTERFACE '92

LIST OF PARTICIPANTS

David M. Allen
University of Kentucky
Department of Statistics
Lexington, KY 40506
allen@tamu.stat.edu

Gerald R. Andersen
Army Research Office
P.O. Box 12211
Research Triangle Park, NC 27709
jerry@brl.mil

Stewart J. Anderson
University of Pittsburgh
302 Arron Hall
Graduate School of Public Health
Pittsburgh, PA 15213
sja@vms.cis.pitt.edu

Matthew R. Arrott
NCSA
605 East Springfield Ave
Champaign, IL 61821
arrott@ncsa.uiuc.edu

Taskin Atilgan
AT&T Bell Laboratories
600 Mountain Ave
Murray Hill, NJ 07974

E. Neely Atkinson
MD Anderson Cancer Center
1515 Holcombe Blvd.
Houston, TX 77030
neely@biomath.mda.uth.tmc.edu

Siu Tong Au
AT&T Bell Laboratories
600 Mountain Ave 7B\517
Murray Hill, NJ 07974
attcom.ulysses!ast

Elton P. Avara
US Army Atmospheric Sciences Lab
5109 Bastille
El Paso, TX 79924

Isam Ayoubi
Texas A&M University
Department of Computer Science
College Station, TX 77843-3112
ayoubi@cs.tamu.edu

Jangsun Baek
SMU
Dept. of Statistical Science
Dallas, TX 75275
baek@lust.csem.smu.edu

Brenda S. Baker
AT&T Bell Laboratories
600 Mountain Ave Room 2C-457
Murray Hill, NJ 07974
bsb@research.att.com

Pierre A. Balthazard
University of Arizona
BPA 406
Tucson, AZ 85721
pbalthazard@mis.arizona.edu

David L. Banks
Carnegie Mellon University
Department of Statistics CMU
Pittsburgh, PA 15213
banks@stat.cmu.edu

Andrew Barron
Barron Associates
Route 1 Box 159
Stanardsville, VA 22973
barron@andrew.stat.uiuc.edu

Austin Barron
American University
Washington, DC 20016
abarron@american.edu

Douglas Bates
University of Wisconsin
1210 W. Dayton Street
Madison, WI 53706
bates@stat.wisc.edu

Ron Baxter
CSIRO Division of Math and Stat
P.O. Box 218
Lindfield, Australia NSW 2112
ronb@syd.dms.csiro.au

Kate Beard
NCGIA/University of Maine
Department of Survey Engineering
Orono, ME 04455

Richard A. Becker
AT&T Bell Laboratories
600 Mountain Ave. Room 2C259
Murray Hill, NJ 07974
rab@research.att.com

Lynne Billard
University of Georgia
Department of Statistics
Athens, GA 30602
statuga@uga

Andrew Black
Harris Semiconductor
MS 62-24 P.P. BOX 803
Melbourne, FL 32901
ablack@msl6.mis.semi.harris.com

Peter Blaskiewicz
Baylor University
Department of Math
Waco, TX 76798

James Blinn
Caltech
305 S. Hill
Pasadena, CA 91106
blinn@gumby.cs.caltech.edu

Mary Ellen Bock
Purdue University
Department of Statistics
1399 Math. Bldg
West Lafayette, IN 47907-1399
mbock@l.cc.purdue.edu

Barry Bodt
U.S. Army Ballistic Research Lab.
Director USABRL
Attn. SLCBR-SE (Bodt)
APG, MD 21005-5066
babodt@smoke.brl.mil

Andrew Booker
Boeing Computer Services
P.O. Box 24346 MS 7L-21
Seattle, WA 98124-0346
booker@espresso.boeing.com

Jane M. Booker
Los Alamos National Lab
Statistics Group MS F600
Los Alamos, NM 87545
jmb@lanl.gov

Kimberly Boomer
Wheaton College
Wheaton Box W0243
Norton, MA 02766
kboomer@wheatnma.bitnet

Kenneth P. Bowman
University of Illinois
Dept. of Atmos. Sciences
105 S. Gregory Ave
Urbana, IL 61801
bowman@uiatma.atmos.uiuc.edu

Leo Breiman
UC Berkeley
Dept. of Statistics
Berkeley, CA 94720
leo@stat.berkeley.edu

Ronald Bremer
Texas Tech University
College of Business Administration
Lubbock, TX 79409-2101
odron@ttacs

Bradford S. Brown
Quality & Statistics
1803 Portsmouth Street
Houston, TX 77048

Robert Brunell
Southern Methodist University
Dept. of Statistical Science
Dallas, TX 75275
h9rr1001@smuvml.bitnet

Judith A. Buchino
1307 Leighton Circle
Louisville, KY 40222

David A. Burn
Minitab Inc.
3081 Enterprise Drive
State College, PA 16801

Judith A. Calem
WSS
4601 Connecticut Ave
Washington, DC 20008

James A. Calvin
Texas A&M University
Department of Statistics
College Station, TX 77843-3143
calvin@stat.tamu.edu

Daniel M. Cap
BFGoodrich Company
9921 Breckville Road
Brecksville, OH 44141

Vincent Carey
Johns Hopkins University
4 Upland Road #12
Baltimore, MD 21210
vjcarey@sphunix.sph.jhu.edu

Brad Carlin
University of Minnesota
Division of Biostatistics
School of Public Health Mayo Memor
Minneapolis, MN 55455
brad@muskie.biostat.unm.edu

Daniel B. Carr
George Mason University
Center for Computational Statistics
Fairfax, VA 22030
dcarr@galaxy.gmu.edu

Hassan Charara
Texas A&M University
Department of Computer Science
College Station, TX 77844
hassan@visualz.cs.tamu.edu

Chantal Martine Chauvelier
Texas A&M University
Department of Geophysics
College Station, TX 77843-3114

Cheng Cheng
Texas A&M University
Department of Statistics
College Station, TX 77843-3143
cheng@stat.tamu.edu

Omar Cherkaoui
Université de Québec
CP 8888 Suc A
Montreal, Quebec M3C 1P1
cherkaou@meki.info.uqam.ca

Robert M. Chervin
National Center for Atmospheric Research
P.O. Box 3000
Boulder, CO 80301
chervin@ncar.ucar.edu

Bart Childs
Texas A&M University
Department of Computer Science
College Station, TX 77843-3112
bart@cs.tamu.edu

I-Shang Chou
University of California
Department of Statistics
Riverside, CA 92521
chou@ucr.stat.ucr.edu

Cindy Christiansen
University of Texas
Department of Mathematics
Austin, TX 78712
cindy@math.utexas.edu

Hyung-Min (Michael) Chung
Texas A&M University
BANA
College Station, TX 77843
hmchung@tamcba

K.W. Church
AT&T Bell Laboratories
600 Mountain Ave
Murray Hill, NJ 07974
kwc@research.att.com

Linda Clark
AT&T Bell Laboratories
600 Mountain Avenue Room 2C-273
Murray Hill, NJ 07974
lac@research.att.com

Daren B.H. Cline
Texas A&M University
3112 Rolling Glen
Bryan, TX 77801
dcline@stat.tamu.edu

Michael Conlon
University of Florida
Box J212 JHMC
Gainesville, FL 32610
mcolon@stat.ufl.edu

Margaret C. Conomos
U.S. Environmental Protection Agency
401 M Street, S.W. (TS-798)
Washington, DC 20460
FAX 202 260-0018

Dianne H. Cook
Rutgers University
Department of Statistics
Hill Center Busch Campus
New Brunswick, NJ 08903
dcook@stat.rutgers.edu

Dennis Cox
University of Illinois
Department of Statistics
101 Illini Hall
725 S. Wright Stree
Champaign, IL 61820
dcox@vmd.cso.uiuc.edu

Giles L. Crane
NJ State Department of Health
73 Philip Drive
Princeton, NJ 08540
Home Phone 609 924-0971

Jonathan D. Cryer
University of Iowa
Department of Statistics
Iowa City, IA 52242
jcryer@stat.uiowa.edu

Carl Davis
IASC
Central Statistics Office
Government Buildings, Cardiff Road
Newport, Gwent, UK NP9 1XG
010 44 633 812863

Lorraine Denby
AT&T Bell Laboratories
600 Mountain Ave Rm 2C255
Murray Hill, NJ 07974
ld@research.att.com

Jack Denny
University of Arizona
Department of Statistics
Tucson, AZ 85721
denny@arizrvax

Kim-Anh Do
Australian National University
Centre for Maths & Its Applns
Canberra, Australia ACT 2601
dokstat@durra.anu.edu.au

David A. Dryer
U.S. Army
1120 Leahy Road
Monterey, CA 93940
trac@cc.nps.navy.mil

Bonnie P. Dumas
Westraco - Forest Research
P.O. Box 1950
Summerville, SC 29484

Brian Duston
Naval Ocean Systems Center
271 Catalina Blvd
San Diego, CA 92152
duston@nosc.mil

Mark Eakin
UT Arlington
P.O. Box 19437
Arlington, TX 76019-0437
b118mee@utarlvm1

William F. Eddy
Carnegie Mellon University
Department of Statistics
Pittsburgh, PA 15213
bill@stat.cmu.edu

Sam Efromovich
University of New Mexico
Dept. Math. Stat.
Hum Brdg 415
Albuquerque, NM 87131
efrom@gopher.unm.edu

Stephen G. Eick
AT&T Bell Laboratories
600 Mountain Ave
Naperville, IL 60566
eick@research.att.com

John F. Elder IV
University of Virginia
Dept. of Systems Engineering
Charlottesville, VA 22903-2442
jfe7w@virginia.edu

Babiker Elmamoun
Texas A&M University
Department of Computer Science
College Station, TX 77840
babiker@cs.tamu.edu

Elizabeth Eltinge
Texas A&M University
Department of Statistics
College Station, TX 77843-3143
beth@stat.tamu.edu

John L. Eltinge
Texas A&M University
Department of Statistics
College Station, TX 77843-3143
jeltinge@stat.tamu.edu

Laszlo Engelman
Systat Inc.
1800 Sherman Ave.
Evanston, IL 60201

Thomas W. Epps
University of Virginia
Economics Department Rouss Hall
Charlottesville, VA 22901

Marian Eriksson
Texas A&M University
Dept. of Forest Science
College Station, TX 77843-
marian@regis3.tamu.edu

Alaattin Erkanli
Duke University
ISDS, Duke University
Durham, NC 27705
ae@isds.duke.edu

Randy Eubank
Texas A&M University
Department of Statistics
College Station, TX 77843-3143
e415re@tamvm

William Evanco
MITRE
4525 Colshire
McLean, VA

Michael Evans
University of Toronto
Department of Statistics
Toronto, Ontario M5S1A1
mevans@utstat.toronto.edu

Richard A. Faldowski
University of North Carolina
Dept. of Psychology
Davies Hall CB 3270
Chapel Hill, NC 27514
faldowra@vanderbilt.crvax.edu

Julian J. Faraway
University of Michigan
Department of Statistics
Ann Arbor, MI 48109
julian@stat.isa.umich.edu

Don R. Faxon
George Mason University
Center for Comp. Stat.
Fairfax, VA 22030
dfaxon@gmuva

Jean-Michel Fayolle
Texas A&M University
430 Southwest Pkwy Apt. 1820
College Station, TX 77840
fayolle@stat.tamu.edu

Mark A. Fleming
Texas A&M University
Department of Computer Science
College Station, TX 77843-3112
markf@cs.tamu.edu

William F. Fulkerson
Deere & Company-Technical Center
3300 River Drive
Moline, IL 61265
435-5311@mczmail.com

Mark Gaither
Texas A&M University
Department of Computer Science
College Station, TX 77843-3112
markg@cs.tamu.edu

Lionel Galway
RAND
1700 Main Street
Santa Monica, CA 90407
galway@rand.c.g

William V. Garetz
US Environmental Protection Agency
1413 Fifth Street Northwest
Washington, DC 20001

John M. Garner
S.F. Austin State University
4097 Sandalwood
Nacogdoches, TX 75961
jmgarner@sfaustin.bitnet

Alan E. Gelfand
University of Connecticut
Storrs, CT 06269
gelfand@uconnvm.bitnet

Andrew Gelman
UT Berkeley
Department of Statistics
Berkeley, CA 94720
gelman@stat.berkeley.edu

James E. Gentle
IMSL Inc.
4141 Southwest Frwy
Sugarland, TX 77478
gentle@imsl.com

Robert Gentleman
University of Waterloo
Department of Statistics
Waterloo, Ontario N2L 3G1
rgentlem@ea1stat.uwaterloo.ca

Alexander A. Georgiev
Ethyl Corporation
P.O. Box 14799
Baton Rouge, LA 70898

Jack Gerson
VA Medical Center/UC San Francisco
4150 Clement 116R
San Francisco, CA 94121
jack@stat.berkeley.edu

Martin Gilchrist
Springer-Verlag
3600 Pruneridge Ave Suite 200
Santa Clara, CA 95051
gilchris@sccm.stanford.edu

Wally Gilks
MRC Biostatistics Unit
5 Shaftesbury Road
Cambridge, U.K. CB2 2BW
wally@mrc-bsu.cam.ac.uk

Sheri D. Gilley
SPSS Inc.
444 North Michigan Ave
Oak Park, IL 60304
sher@spss.com

Michael F. Goodchild
UC Santa Barbara
Department of Geography/NCGIA
Santa Barbara, CA 93106-4060
good@ncgia.ucsb.edu

Arnold Goodman
County of Los Angeles
18231 Hillcrest Circle
Villa Park, CA 92667

Paul Gregor
VA Medical Center
1911 Harold Street
Houston, TX 77098

Alan M. Gross
Bellcore
70 Stanwick Court
Somerset, NJ 08873
amg@bellcore.com

Eunho Ha
Texas A&M University
Department of Statistics
College Station, TX 778432-3143
ha@stat.tamu.edu

Mark S. Handcock
New York University
90 Trinity Place #1010
New York, New York 10006-1594
handcock@nannup.stern.nyu.edu

Brad Hanson
American College Testing
P.O. Box 168
Iowa City, IA 52243
bhanson@umaxa.weeg.uiowa.edu

J. Michael Hardin
University of Alabama
720 S. 20th St.
Tidwell Hall Room 212
Birmingham, AL 35294

James Hardin
Texas A&M University
Department of Statistics
College Station, TX 77843-3143
jhardin@stat.tamu.edu

Janis Hardwick
University of Michigan
3340 Fernwood Ave.
Ann Arbor, MI 48108
janis.hardwick@vb.ub.cc.umich.edu

Jeffrey Hart
Texas A&M University
Department of Statistics
College Station, TX 77843-3143
hart@stat.tamu.edu

Gary Haskin
Boots Pharmaceuticals
8800 Ellerbe Road P.O. Box 6750
Shreveport, LA 71106

Trevor Hastie
AT&T Bell Laboratories
600 Mountain Ave. 2C 261
Murray Hill, NJ 07974
trevor@research.att.com

Leonard B. Hearne
George Mason University
1829 E. Capitol St. SE
Washington, DC 20003

Richard M. Heiberger
Temple University
Department of Statistics
Philadelphia, PA 19122-2585
rmh@astro.ocis.temple.edu

J.I. Helfman
AT&T Bell Laboratories
600 Mountain Ave.
Murray Hill, NJ 07974
jon@research.att.com

Joe R. Hill
EDS Research
5951 Jefferson St. NE
Albuquerque, NM 87109
joe@edsr.eds.com

Mary Ann Hill
Systat Inc.
1800 Sherman Ave.
Evanston, IL 60201

Ronald Hocking
Texas A&M University
Department of Statistics
College Station, TX 77843-3143
hocking@stat.tamu.edu

Lorrie L. Hoffman
University of Central Florida
Department of Statistics
Orlando, FL 32816
fdhoffmn@flvm.cc.ucf.edu

David Holiday
UT Health Center
P.O. Box 2003
Tyler, TX 75710

John J. Hsieh
University of Toronto
12 Queens Park Cr. W. 4th Floor
Toronto, Ontario M5S 1A8

Tailen Hsing
Texas A&M University
Department of Statistics
College Station, TX 77843-3143
thsing@stat.tamu.edu

Fred Hulting
Alcoa Technical Center
D-AMCT, 100 Technical Drive
Alcoa Center, PA 15069
hulting-fl@alcoa.com

Ken Hung
Western Washington University
Parks Hall 213
Bellingham, WA 98225
hung@nessie.cc.wvu.edu

Philip W. Iversen
Iowa State University
117 Snedecor Hall
Department of Statistics
Ames, IA 50011
piversen@iastate.edu

R.K. Jain
Memorial University
Dept. of Math. & Stat.
St. John's NFLD, Canada A1C5S7

Sudha Jain
University of Toronto
1 Fountainhead Rd. # 2104
North York, Ontario M3J 1K6

Rafael Jara
AT&T Bell Laboratories
20 Bayberry Drive
Somerset, NJ 08873
ulysses!rj

Mingda Jiang
IEEE Computer Society
810 W. Sussex Ave.
Missoula, MT 59801
mj@selway.unit.edu

Don Johns
Eli Lilly and Company
Lilly Corporate Center DC 2233
Indianapolis, IN 46285

Mark A. Johnson
Upjohn Laboratories
301 Henrietta Street
Kalamazoo, MI 49001

Iain M. Johnstone
Stanford University
Department of Statistics
Sequoia Hall
Stanford, CA 94305-4065
imj@stat.stanford.edu

Lawrence Jones
Computing Services
State University of New York
Binghamton, NY 13901-6000
ljones@bingvmb

Stephen Kaluzny
Statistical Sciences Inc.
1700 West Lake North Suite 500
Seattle, WA 98109
spk@statsci.com

Robert (Rob) E. Kass
Carnegie Mellon University
1815 Butler Ave. #117
Los Angeles, CA 90025
kass@stat.cmu.edu

Sallie Keller-McNulty
Kansas State University
Department of Statistics
Manhattan, KS 66506

Kenneth E. Kemp
Kansas State University
Department of Statistics
Manhattan, KS 66502
kekemp@ksu.ksu.edu

William G. Kemple
Naval Postgraduate School
CODE OR/KE
Monterey, CA 93943-5000
5222P@navpgs.bitnet

John (Jed) Kenna
Baylor College of Medicine
One Baylor Plaza
Houston, TX 77030
jkenna@bcm.tmc.edu

Kathryn F. Kennedy
UT M.D. Anderson
661 Bering Dr. #102
Houston, TX 77057

David Kennon
The Fair Isaac Companies
120 North Redwood Drive
San Rafael, CA 94903

Myrna M. Khan
Baylor College of Medicine
Computing Resources Center
Irel 3rd Floor
Houston, TX 77030-3498

John King
Burroughs Wellcome Co.
3030 Cornwallis Road
Research Triangle Park, NC 27709

Anita Klein
National Crop Insurance Services
7301 College Blvd.
Overland Park, KS 66210

Charles Koelbel
Rice University
6100 S. Main
Houston, TX 77251
chk@rice.edu

Roger Koenker
University of Illinois
504 W. Pennsylvania
Urbana, IL 61820
roger@ysidro.econ.uiuc.edu

Andrzej S. Kozek
University of Arkansas
705 Putman R3
Fayetteville, AR 72201
akozek@vafsysb

Shenghan Lai
University of Miami
Comprehensive AIDS Program
School of Medicine
1800 NW 10th Ave.
Miami, FL 33136

Margaret Land
Texas A&I University
Campus Box 172
Kingsville, TX 78363-8201

Stephanie Land
Stanford University
Department of Statistics
Sequoia Hall
Stanford, CA 94305-4065
steph@playfair.stanford.edu

James M. Landwehr
AT&T Bell Laboratories
Room 2C-257
Murray Hill, NJ 07974-0636
jml@research.att.com

Kinley Larntz
University of Minnesota
Department of Applied Statistics
352 Claoff Bldg
St. Paul, MN 55108
kinley@umnstat.stat.umn.edu

Edmund Lau
Failure Analysis Associates
149 Commonwealth Drive
Menlo Park, CA 94025

Tommy W. Lee
UT San Antonio
110 Somerset Drive
San Antonio, TX 78211-1910
tlee@ringer.cs.utsa.edu

Tze-San Lee
Western Illinois University
Department of Mathematics
900 W. Adams Street
Macomb, IL 61455
mftl@ecnuxa.bitnet

Kevin Leonard
Wilfrid Laurier University
75 University Ave. West
Waterloo, Ontario NZL 3C5

Raoul LePage
Michigan State University
Department of Statistics and Probability
East Lansing, MI 48824
rdl@lepage-sun.stt.msu.edu

Ming-Ying Leung
University of Texas
Division of Mathematics
San Antonio, TX 78249-0664
mcmyl@utsa86.sa.utexas.edu

Peter A. Lewis
Naval Postgraduate School
Box 5444
Monterey, CA 93941
1526@navpgs

Keh-Shin Lii
University of California
Riverside, CA 92521
ksl@ucrstat.ucr.edu

Foster Lindley
University of Connecticut
32 Ledgewood Drive
Storrs, CT 06268
vpacad20@uconnvms

Michael Littman
Bellcore Cognitive Sc. Research Gr.
445 South Street Room 2L-331
Morristown, NJ 07962-1910
mlittman@bellcore.com

Shin Ta Liu
Northern Telecom
16350 W. Bernardo Drive
San Diego, CA 92127

Michael Lloyd
University of Heriot-Watt
Dept. of Act. Math & Stats
Riccarton Edinburgh, Scotland EH14 4AS
mike@cara.ma.hw.ac.uk

Michael T. Longnecker
Texas A&M University
3109 Hummingbird Circle
Bryan, TX 77801
longneck@stat.tamu.edu

Zhiyi Lou
Montreal Children's Hospital
4060 Ste-Catherine St. West
Montreal, Quebec PQ H3Z 2Z3
lou@homer.cs.mcgill.ca

Qiang Luo
George Mason University
157 Science-Technology Bldg II
4400 University Drive
Fairfax, VA 22030
qluo@galaxy.gmu.edu

Margaret S. Mackisack
Queensland University of Tech
1609 Pleasant Street #113
Lauderdale, MN 55108
mackisack@gut.edu.au

David J. Marchette
Naval Ocean Systems Center
Code 421
San Diego, CA 92152
marchett@nosec.mil

David M. Mark
NCGIA/SUNY At Buffalo
SUNY/Buffalo
Dept. of Geography
Buffalo, NY 14261
in%geodmm@ubvms.bitnet

Paul L. Marsh
North Carolina State University
Department of Statistics Box 8203
Raleigh, NC 27695-8203
marsh@stat.ncsu.edu

Ron McCright
SMU
215 Sam Drive
Waco, TX 76706

Ennis Donice McCune
Stephen F. Austin State University
Box 13040 SFA
Nacogoches, TX 75962

Allen McIntosh
Bellcore
445 South Street Room 2Q-324
Morristown, NJ 07960-1910
mcintosh@bellcore.com

John D. McKenzie
Babson College
Babson Park
Wellwlesley, MA 02157
mckenzie@babson

A. Ian McLeod
University of Western Ontario
Dept. of Stat. & Acturial Sciences
Paterson Bldg. Room 284
London, Ontario, Canada N6A 5B7
aim@stats.uwo.ca (FAX 661-3813)

Nancy McMahon-Cox
SPSS Inc.
444 North Michigan Ave
Oak Park, IL 60304
sheri@spss.com

Cyrus R. Mehta
Harvard University
137 Erie Street
Cambridge, MA 02139
mehta@jimmy.harvard.edu

Sunil Menon
Texas A&M University
Dept. of Nuclear Engineering
College Station, TX 77843-3133
skm1526@zeus.tamu.edu

Michael M. Meyer
Carnegie Mellon University
Pittsburgh, PA 15213
mikem@stat.cmu.edu

Ted W. Mihalisin
Temple University
600 Honey Run Road
Ambler, PA 19002

Charles E. Miller
TRAC-WSMAR
ATRC-WMA
White Sands Missile Range, NM 88002

Michael C. Minnotte
Rice University
P.O. Box 1892
Houston, TX 77251
minnotte@rice.edu

John Monahan
North Carolina State University
Department of Statistics
Raleigh, NC 27695-8203
monahan@stat.ncsu.edu

Mark Monmonier
Syracuse University
302 Waldorf University
Syracuse, NY 13224
monzier@sunrise.acs.syr.edu

Leslie M. Moore
Los Alamos National Laboratory
MS F600
Los Alamos, NM 87545
lmmy@lanl.gov

Dale Morgan
Texas A&M University
Dept. of Geophysics
College Station, TX 77843

Carl Morris
Harvard University
Department of Statistics
Science Center 1 Oxford Street
Cambridge, MA 02138
morris@morris.harvard.edu

Sally C. Morton
RAND
1700 Main Street
Santa Monica, CA 90407-2138
sallymorton@rand.org

Peter Mueller
Duke University
ISDS, Duke University
Durham, NC 27705
pm@ids.duke.edu

Matoteng M. Ncube
University of West Florida
1100 University Pkwy
Pensacola, FL 32514

Padraic Neville
18 Prospect Ave., Box 114
Port Coast, CA 94569
neville@stat.berkeley.edu

Michael Newton
University of Wisconsin
1210 West Dayton Street
Madison, WI 53706
newton@stat.wisc.edu

Katherine Ng
SAS
SAS Campus Drive
Cary, NC 27513
ng@unx.sas.com

Pin Ng
University of Houston
Houston, TX 77204-5882
pin@menudo.uh.edu

Xufeng Niu
Florida State University
Department of Statistics
Tallahassee, FL 32306
niu@stat.fsu.edu

John Nolan
American University
Dept. of Math.
4400 Massachusetts Ave. N.W.
Washington, DC 20016
jpnolan@american.edu

Gerald R. North
Texas A&M University
Department of Meteorology
College Station, TX 77843

David R. Olson
Centers for Disease Control
1600 Clifton Road
Atlanta, GA 30333

John Ondrasik
Astra Pharmaceutical Products Inc.
Westborough, MA 01581

George Ostroughov
Oak Ridge National Laboratory
P.O. Box 2008 Bldg 6012
Oak Ridge, TN 37931-6367
ost@msr.epm.ornl.gov

Donna Palski
Baylor University
909 Baylor #403
Waco, TX 76706-

Panickos Palettas
Virginia Tech
Dept. of Statistics
Blacksburgh, VA 24061

Judy L. Palmer
UT M.D. Anderson Cancer Center
Dept. of Biomathematics
1515 Holcombe Blvd
Houston, TX 77030
jlp@odin.mda.uth.tmc.edu

Young Ryong Park
Syracuse University
B18 Slocum Heights Apt. #1
Syracuse, NY 13210
young@suvn.bitnet

Emanuel Parzen
Texas A&M University
Department of Statistics
College Station, TX 77843-3143
eparzen@stat.tamu.edu

Michael Parzen
Harvard University
Department of Biostatistics
Cambridge, MA 02146
parzen@jimmy.harvard.edu

Ranjit M. Passi
Institute for Naval Oceanography
Stennis Space Center, MS 39529
@ucan.ino.edu

Gary L. Patton
Radian Corporation
P.O. Box 201088
Austin, TX 78720-1088
radian!gary@natinot.com

Mirek Pawlak
University of Manitoba
Dept. of Electr. and Comp. Eng.
Winnipeg, Manitoba R3T2NC
pawlak@eeserv.ee.umanitoba.ca

Don Percival
University of Washington
HN - 10
Seattle, WA 98105
dbp@apl.washington.edu

Kathleen G. Perez-Lopez
George Mason University
5881 6th Street
Falls Church, VA 22041
klopez@gmuvax.gmu.edu

M. Petri
AT&T Bell Laboratories
600 Mountain Ave
Murray Hill, NJ 07974

Krzysztof Podgorski
Michigan State University
Department of Probability and Statistics
East Lansing, MI 48824
pdo@stt3b2.stt.msu.edu

Andy Podgurski
Computer Eng. & Science Dept.
Case Western Reserve University
Crawford Hall 10900 Euclid Ave
Cleveland, OH 44106
andy@alpha.ces.cwru.edu

Vincent Poor
Electrical Engineering Department
Princeton, NJ 08544
poor@ivy.princeton.edu

Daryl Pregibon
AT&T Bell Laboratories
600 Mountain Ave
Room 2C-264
Murray Hill, NJ 07974
daryl@research.att.com

Bonnie K. Ray
Naval Postgraduate School
Monterey, CA 93943
5348p@cc.nps.navy.mil

Lidia Rejtö
University of Delaware
Department of Statistics
Newark, DE 19716
rejtö@chopin.udel.edu

S. Denise Rhodes
Texas A&M University
3309 Robinson #810
Waco, TX 76700
rhodes@stat.tamu.edu

Kate Roach
John Wiley & Sons
605 Third Ave
New York, NY 10158-0012
roach@wiley.comperveuc.com

David G. (Dave) Robinson
AF Institute of Tech IENY
WPAFB
Dayton, OH 45433

Jorge Luis Romeu
201 Rugby Rd.
Syracuse, NY 13203-1440
jromeu@suvvm.bitnet@cunyvnm.cuny.edu

James L. Rosenberger
Penn State University
Department of Statistics
University Park, PA 16801
jlr@stat.psu.edu

Carl T. Russell
U.S. Army OPTEC
Attn. CSTE-ECS-A4501 Fort Avenue
Alexandria, VA 22302
russell@ote2.leav-emh.army.mil

Ed Russell
Advanced Micro Devices
5900 E. Ben White Blvd
Austin, TX 78741

Rami B Safadi
Olin Research Center
350 Knotter Drive
Cheshire, CT 06410
7637.3104@compserve.com

John P. Sall
SAS Institute
SAS Campus Drive
Cary, NC 27513
sall@sas.com

Henrik Schmiediche
Texas A&M University
1600 Welsh 331
College Station, TX 77840
henrik@stat.tamu.edu

Eugene F. Schuster
UT El Paso
Department of Math. Sciences
El Paso, CA 79968
gene@mathep.utexas.edu

David W. Scott
Rice University
Department of Statistics
P.O. Box 1892
Houston, TX 77251
scottdw@rice.edu

Robert Serfling
Johns Hopkins University
Department of Math. Sciences
Baltimore, MD 21218
serfling@jhunix

Kathy Shelley
Iowa State University
3454 Southdale Drive
Ames, IA 50010
kathy@vincent.iastate.edu

David M. Shera
ASA
428 Broadway # 3
Somerville, MA 02145
shera@psych.mgh.harvard.edu

Chen-Chi Shing
Radford University
Computer Science Department
Box 6933
Radford, VA 24142
cshing@yucs.faculty.cs.runet.edu

Jing Shyr
SPSS Inc.
444 N. Michigan Ave
Chicago, IL 60611

Arthur Silverberg
American Cynamid
15 Kensington Ct.
Princeton, NC 08540
uunet!nmri!silverbe

William Smith
Texas A&M University
1040 Rose Circle
College Station, TX 77840
smith@stat.tamu.edu

Paul N. Somerville
University of Central Florida
Orlando, FL 32816
somer@eola.cs.ucf.edu

Roger Sorrells
Texas A&M University
4200 Maywood
Bryan, TX 77801
x041rs@tamvm1.tamu.edu

Paul Speckman
University of Missouri-Columbia
Department of Statistics
Columbia, MO 65211
statpls@umcvm1.bitnet

Phil Spector
UC Berkeley
Department of Statistics
Berkeley, CA 94720
spector@stat.berkeley.edu

Horace J. Spencer
Stephen F. Austin State University
1528 E. Austin #4
Nacogdoches, TX 75961

Scott Spencer
Fingerhut Corp.
4400 Baker Road
Minnetonka, MN 55343

Clifford Spiegelman
Texas A&M University
Department of Statistics
College Station, TX 77843-3143
cliff@stat.tamu.edu

Cathie Spino
Harvard School of Public Health
677 Huntington Ave
Boston, MA 02115
spino@biostat.harvard.edu

S.J. Steel
Potchefstroom University
Department of Statistics
Potchefstroom 2520, South Africa
stkss@pukvm1.puk.ac.za

Elizabeth Stephenson
Becton-Dickison Immunocytometry Sys
2350 Qume Drive
San Jose, CA 95131-1807

Gary Stevens
Sterling Winthrop Pharmaceuticals R
81 Columbia Turnpike
Rensselaer, NY 12144

Maxwell B. Stinchombe
UC San Diego
Department of Economics
La Jolla, CA 92093-0508

Geir Storvik
Norwegian Computing Center
P.O. Box 114, Blindern
Oslo 3, Norway N -0314
geir.storvik@nr.no

Quentin F. Stout
University of Michigan
EECS Department
Ann Arbor, MI 48109-2122
qstout@eecs.umich.edu

Alistair Sutherland
University of Strathclyde
26 Richmond Street
Glasgow, U.K. G1 1NP
a.i.sutherland@vaxa.strath.ac.uk

Virginia Kay Sutton
Baylor University
Department of Math
Waco, TX 76798

Tim Swartz
Simon Fraser University
Department of Math/Stats, SFU
Burnaby, BC-Canada V5A1S6
tim@cs.sfu.ca

Deborah F. Swayne
ASA
Bellcore MRE 2L331 445 South Street
Morristown, NJ 07962-1910
dfs@bellcore.com

William F. Szewczyk
National Security Agency
Attn. R513
Ft. George G. Meade, MD 20755-6000
wfszewc@afterlife.ncsc.mil

Evangelos Tabakis
MIT
Rm 2-333 Dept. of Math.
Cambridge, MA 02139
taba@math.mit.edu

Kwok-Yam Tam
Texas A&M University
Department of Computer Science
College Station, TX 77840-1557
kytam@cs.tamu.edu

Thomas G. Tape
University of Nebraska
600 South 42nd Street
Omaha, NE 68198-3331
tgjape@unmcum

Malcolm Taylor
U.S. Army Ballistic Research Lab.
Director USABRL Attn. SLCBR-SE (Taylor)
APG, MD 21005-5066
mtaylor@smoke.brl.mil

Robert F. Teitel
ABT Associates Inc.
4800 Montgomery Lane Suite 500
Bethesda, MD 20814

Aush M. Thaker
2298 Pheasant
Hercules, CA 94547

Terry M. Therneau
Mayo Foundation
200 First Street. SW GU 1021B
Rochester, MN 55905
therneau@mayo.edu

Eugene W. Thomas
Exxon Exploration Co.
3507 Village Oaks
Kingwood, TX 77339

Ronald G. Thomas
VA Medical Center
3801 Miranda Ave MS 151-K
Palo Alto, CA 94304
rthomas@osiris.stanford.edu

Georgia Thompson
Southern Methodist University
Dallas, TX 75275
h9sr1001@smuvm1

David J. Thomson
AT&T Bell Laboratories
2C-360
Murray Hill, NJ 07974
djt@research.att.com

Herbert Thorwald, Jr.
Texas A&M University
1501 Harvey Road Apt. 644
College Station, TX 77840
kii2cb@tamvm1

John Timar
ASA
Polysar Rubber Co.Box 3001
Sarnia, Ontario, C N7T 7M2
Fax 519 339-7769

Dag Tjøstheim
University of Bergen
Department of Mathematics
Bergen, Norway 5007
dagt@mi.uib.no

Alicia Toledano
Harvard School of Pub. Health
Department of Biostatistics (7th Floor)
677 Huntington Ave.
Boston, MA 02115
toledano@biostat.harvard.edu

Marietta Joan Tretter
Texas A&M University
Business Analysis & Research
College Station, TX 77843-4217
tretter@tamcba

Edward Tufte
Yale University
Box 430
Cheshire, CT 06410

Danny W. Turner
Baylor University Math. Dept.
10004 Stony Point Drive
Waco, TX 76712
turner@baylor.bitnet

Gilbert Walter
University of Wisconsin
Dept. of Mathematics Box 413
Milwaukee, WI 53201
ggw@a.math.uwm.edu

Chung-Ching Wang
University of Central Florida
Orlando, FL 32708

Ferdinand T. Wang
American University
Dept. of Math. and Stat.
4400 Massachusetts Ave. N.W.
Washington, DC 20016-8050
chopin!ferdie@uunet.uu.net

Ruey H. Wang
OLIN Corp.
350 Knotter Drive
Cheshire, CT 06410
75410.3716@compuserve.com

Joe H. Ward
UT San Antonio
167 East Arrowhead Dr.
San Antonio, TX 78228-2402
lmcjhw@utsavm1.bitnet

Harrison Weed
Motorola, Inc.
3501 Ed Bluestein Blvd.
Austin, TX 78721
rwl40@aprdl.sps.mot.com

Edward J. Wegman
George Mason University
Center for Computational Statistics
157 Science-Tech. Bldg.
4400 University Drive
Fairfax, VA 22030
ewegman@endor.galaxy.gmu.edu

Thomas E. Wehrly
Texas A&M University
Department of Statistics
College Station, TX 77843-3143
twehrly@stat.tamu.edu

Robert Weiss
UCLA Dept. of Biostat
School of Public Health
Los Angeles, CA 90024-1772
ibenapk@oac.ucla.edu

Hedy Weissenger
Burroughs Wellcome Co.
3030 Cornwallis Road
Research Triangle Park, NC 27709

Mike West
Duke University
Institute of Stat. and Decision Sciences
Durham, NC 27706
mw@isds.duke.edu

Kerri Whelan
Wheaton College
WI588
Norton, MA 02766
kwhelan@wheatnma.bitnet

Richard E.C. White
Queens College
Academic Computer Center 1121
Flushin, NY 11367
richardw@qcvax

Mary M. Whiteside
UT Arlington
Box 19437
Arlington, TX 76019
b022mmw@utarhvm1

David A. Whitney
The Analytic Sciences Corporation
55 Walkers Brook Drive
Reading, MA 01867
dawhitney@tasc.com

Allan R. Wilks
AT&T Bell Laboratories
600 Mountain Ave., Room 2C-283
Murray Hill, NJ 07974
allan@research.att.com

Matthew Witten
UT CHPC
10100 Burnet Road CMS 1.154
Austin, TX 78712
mwitten@chpc.utexas.edu

John Witz
Texas A&M University
Texas Transportation Institute
College Station, TX 77843-135

Johnny C.S. Wong
AT&T Bell Laboratories
600 Mountain Ave
Room 7C1520
Murray Hill, NJ 07974
attmail:lulysses!jwong

Terry J. Woodfield
Risk Data Corporation
Two Venture Plaza, Suite 400
Irvine, CA 92718-3331

Trong Wu
Southern Illinois University
Computer Science Department
Edwardsville, IL 62026-1656
cp00@siuemus.bitnet

Shenq-Na Yang
UT Richardson
Program in Math. Sciences
P.O. Box 8380688
Richardson, TX 75083-0688
snyang@utdallas.edu

Forrest Young
University of North Carolina
CB 3270
Chapel Hill, NC 27599
uluru@unc.bitnet

Demirhan Yenigun
AT&T Bell Laboratories
600 Mountain Ave Room 7b-511A
Murray Hill, NJ 07974

Mingxian Xu
George Mason University
Center for Computational Statistical
157 Science & Tech. II
4400 University
Fairfax, VA 22030
ming@gmuvax2.gmu.edu

Yun Zhou
Baylor University
1905 S, 8th Street Apt #2
Waco, TX 76706-

Stuart Zweben
Ohio State University
2036 Neil Ave
Columbus, OH 43210
zweben@cis.ohio_state.edu

INDEX OF AUTHORS

Abkowitz, J.L.	449	Eubank, R.L.	1	Larson, H.	282
Agresti, W.W.	333	Evanco, W.M.	333	Lavine, M.	369
Armstrong, E.M.	287	Evans, M.	309	LePage, R.	549
Baker, B.S.	48	Faldowski, R.A.	223	Lee, T.S.	258
Balthazard, P.A.	28	Faraway, J.J.	248	Leonard, K.J.	28
Banks, D.	369	Ferland, R.	103	Leung, Y.	416
Bates, D.M.	456	Fisher, N.	358	Lewis, P.A.W.	297
Baxter, R.	358	Galway, L.	107	Limber, M.	379
Beard, M.K.	408	Gelman, A.	433	Lin, C.C.	416
Becker, R.A.	111	Genz, A.	178	Littman, M.	207
Benedict, J.P.	192	Gerson, J.	253	Lloyd, M.	197
Black, A.J.	125	Ghorai, J.K.	233	Lou, Z.	39
Booker, A.J.	270	Gilks, W.R.	439	Marasinghe, M.G.	120
Bowman, S.	517	Goel, A.L.	344	McFarlane, M.M.	223
Brodkin, C.	581	Goodchild, M.F.	416	Mehta, C.R.	557
Brown, D.E.	568	Goodrich, R.K.	379	Meyer, M.M.	504
Buja, A.	207	Gregor, P.J.	34	Mihalisin, T.	140
Buja, A.	479	Guttorp, P.	449	Mockus, A.	504
Buttenfield, B.P.	408	Hagen, D.	116	Monmonier, M.	423
Cabrera, J.	475	Handcock, M.S.	77	Mueller, P.	317
Cabrera, J.	479	Hardin, J.M.	181	Murray, T.J.	499
Chen, C.	499	Hardin, J.W.	292	Ncube, M.M.	130
Cherkaoui, O.	103	Hardwick, J.P.	595	Newton, M.A.	449
Church, K.W.	57	Hardwick, J.P.	600	Ng, P.T.	384
Ciampi, A.	39	Hearne, L.B.	484	Niu, X.	84
Coley, K.L.	287	Heiberger, R.M.	111	Nolan, J.P.	18
Conlon, M.	464	Helfman, J.I.	57	Park, Y.R.	499
Cook, D.	475	Hoffman, L.L.	397	Parzen, E.	605
Cook, D.	479	Holder, M.W.	287	Passi, R.M.	379
Crane, G.L.	275	Holiday, D.B.	489	Patel, N.R.	557
Dean, N.	207	Hsieh, J.J.	186	Patton, G.L.	287
Denby, L.	243	Hulting, F.L.	159	Paul, R.	344
Do, K.	512	Hung, K.	116	Percival, D.B.	537
Dryer, D.A.	282	Iversen, P.W.	120	Pittard, C.L.	568
Eakin, M.	394	Jain, R.K.	578	Podgorski, K.	549
Eakin, M.	517	Jain, S.	522	Podgurski, A.	349
Eddy, W.F.	504	Jiang, M.	494	Poor, H.V.	542
Efromovich, S.	374	Kass, R.E.	178	Ray, B.K.	297
Eick, S.G.	67	Kemple, W.	282	Relles, D.	107
Elder, J.F.	568	Koenker, R.W.	384	Robinson, D.	581
Elder, J.F.	572	Kokic, P.	512	Romeu, J.L.	304
Elder, J.F.	9	Kozek, A.S.	388	Romeu, J.L.	527
Epps, T.W.	590	Lamont, R.W.	282	Russell, E.L.	169
Erkanli, A.	178	Landwehr, J.M.	243	Safadi, R.B.	135

Sall, J.	217
Schervish, M.J.	504
Schmiediche, H.	292
Schuster, E.F.	532
Schwegler, J.	140
Shing, C.C.	400
Silverberg, A.	202
Singh, K.P.	181
Somerville, P.N.	262
Speckman, P.L.	1
Spector, P.	469
Spiegelhalter, D.J.	439
Storvik, G.	94
Stout, Q.F.	595
Stout, Q.F.	600
Sutherland, A.	403
Swartz, T.	309
Swayne, D.F.	207
Switzer, P.	94
Szewczyk, W.F.	192
Tabakis, E.	585
Tan, K.	504
Teitel, R.F.	44
Templeton, J.G.C.	522
Therneau, T.M.	365
Thomas, A.	439
Thompson, G.L.	149
Tiao, G.C.	84
Timlin, J.	140
Viele, K.	504
Wallace, R.C.	287
Wallis, J.R.	77
Walmsley, M.	358
Walter, G.G.	233
Wang, C.	202
Wang, M.C.	21
Wang, R.H.	135
Wegman, E.J.	484
Weiss, R.E.	265
West, M.	324
Wright, A.H.	494
Wu, T.	24
Yang, K.Z.	344
Young, F.W.	223